Deep RL Inventory Management with Supply and Capacity Risk Awareness

Defeng Liu [1] **Ying Liu** [1] **Carson Eisenach** [1]

Abstract

In this work, we study how to efficiently apply reinforcement learning (RL) for solving large-scale stochastic optimization problems by leveraging intervention models. The key of the proposed methodology is to better explore the solution space by simulating and composing the stochastic processes using pre-trained deep learning (DL) models. We demonstrate our approach on a challenging real-world application, the multi-sourcing multi-period inventory management problem in supply chain optimization. In particular, we employ deep RL models for learning and forecasting the stochastic supply chain processes under a range of assumptions. Moreover, we also introduce a constraint coordination mechanism, designed to forecast dual costs given the cross-products constraints in the inventory network. We highlight that instead of directly modeling the complex physical constraints into the RL optimization problem and solving the stochastic problem as a whole, our approach breaks down those supply chain processes into scalable and composable DL modules, leading to improved performance on large real-world datasets. We also outline open problems for future research to further investigate the efficacy of such models.

1. Introduction

Multi-sourcing and multi-period inventory management problem (MMIMP) is a challenging real-world application in stochastic optimization. When complex stochastic supply chain processes and constraints exist, it is not computationally trackable to directly solving the problem using classic optimization techniques such as dynamic programming. Despite making suboptimal decisions, modern inventory management systems (IMS) in retail supply chain (such as Walmart and Amazon) typically deploy heuristic-based multi-sourcing buying strategies, which employ a combined system with a just-in-time (JIT) ordering strategy plus other specialized strategies that strive to achieve a balance between supply shortage and inventory health across all products without compromising the inventory management's contribution to the overall retail service.

In real world supply chains, order quantities are subject to several post-processors, including modification to meet vendor constraints such as minimum order and batch size constraints. Second, the supply may be unreliable and vendors may only partially fill orders that they receive. This may occur for multiple reasons, including that the vendor itself is out of stock. In the literature the proportion of the original order quantity retailer ultimately receives is referred to as the yield or fill rate. In the current state, there is no existing representation model to learn those external processes.

This work is motivated by the prior work (Madeka et al., 2022) which established the viability of Deep RL for single-sourcing inventory planning problem. But we forward this line of research by targeting for a more complex variant of the aforementioned problem, wherein multi-sourcing channels of vendors are available and the primary trade-off considered was to strike a balance for costs vs supply risks for different vendors. On the other hand, we aim to investigate effects of stochastic quantity over time arrival profiles.

Here, we emphasize that a critical issue encountered in solving the multi-sourcing inventory problem via traditional methods such as dynamic programming is the unknown dynamics of a variety of underlying processes associated with inventory control. For instance, customer demand is not deterministic and exhibits volatility which are influenced by seasonality, external sourcing processes, etc. To incorporate those complex supply chain processes, we attempt to investigate whether it is possible to efficiently scale the training of decision policies from the vast amount of data generated from intervention models for the various state variables with stochastic behaviors.

We organize rest of the paper as summarized next. In Section 2, we mathematically formulate the dual-sourcing inventory management problem and thereby describe our dual sourcing RL methodology as a solution for the problem.

[1]Amazon. Correspondence to: Defeng Liu <liudef@amazon.com>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

In Section 3, we present our main results from numerical experiments. Additionally, in Section 4, we extend our dual sourcing RL baseline model by introducing a capacity mechanism control strategy and provide evaluation results. Finally, we conclude the paper with discussion on future work in Section 5.

2. A Deep RL Approach for the Dual Sourcing Problem

2.1. Modeling the Dual Sourcing Problem as an Exo-IDP

In this section we model the dual sourcing problem as an *Exogenous Interactive Decision Process* (Exo-IDP). We consider the case of a retailer managing a set $\mathcal{A}$ of products for T time steps, where the objective is to maximize revenue by placing orders to both *long lead time* (LLT) and JIT sources. To succinctly describe our process, we focus on just one product $i \in \mathcal{A}$, though we note that decisions can be made jointly for every product.

State The price received at sale, costs incurred on JIT purchase, and holding costs are denoted as p_t^i, $c_t^{J,i}$, and h_t^i, respectively. For LLT orders, the retailer typically receives a discount on the cost of goods sold, so there is a different cost incurred on purchase $c_t^{L,i}$. , Additionally, the demand for product i at time t is denoted as d_t^i. The aforementioned set of variables are completely exogenous, and therefore their evolution is independent of any policy's interaction with the Exo-IDP.

Together these exogenous processes form the state as follows,

$$s_t^i \triangleq \big(d_t^i, p_t^i, c_t^{J,i}, c_t^{L,i}, , \boldsymbol{\rho}_t^{J,i}, \boldsymbol{\rho}_t^{L,i}, \boldsymbol{M}_t^{J,i}, \boldsymbol{M}_t^{L,i}\big). \quad (1)$$

The history of the joint process up to time t is defined as

$$H_t := \{(k_0^i, s_1^i, \ldots, s_t^i)\}_{i=1}^{|\mathcal{A}|},$$

where k_0^i is the initial inventory level. Product-level histories can be defined similarly.

Actions For product i, action a_t^i implies placing orders via JIT, LLT channels at time t. In other words, agent's interaction with the Exo-IDP is only via placing orders. More precisely,

$$a_t^i \triangleq (q_t^{J,i}, q_t^{L,i}). \quad (2)$$

So we have $a_t^i \in \mathbb{R}^2$. For a class of policies parameterized by θ, we can defined the actions as

$$a_t^i = \pi_{\theta,t}^i(H_t). \quad (3)$$

We define the set of these policies as $\mathbf{\Pi} \triangleq \{\pi_t^i(;\theta)|\theta \in \Theta, i \in \mathbb{A}, t \in [0, T-1]\}$.

External Sourcing Processes After orders are created and submitted to vendors, they can arrive in multiple shipments over time, and the total arriving quantity may not necessarily sum up to the order quantity placed. Any constraints the vendor imposes on the retailer's orders $\boldsymbol{M}_t^{J,i} \in \mathbb{R}^{d_v}$ – such as minimum order quantities and batch sizes – are exogenous to the ordering decisions. We define an order quantity post-processor on the JIT source $f_p^J : \mathbb{R}_{\geq 0} \times \mathbb{R}^{d_v} \to \mathbb{R}_{\geq 0}$ that may modify the order quantity. The final order quantity submitted to the vendor is denoted as $\tilde{q}_t^{J,i} := f_p^J(q_t^{J,i}, \mathrm{M}_t^{J,i})$. Similarly, the final LLT order quantity is defined as $\tilde{q}_t^{L,i} := f_p^L(q_t^{L,i}, \mathrm{M}_t^{L,i})$.

At every time t, the vendor has allocated a supply $U_t^{J,i}$ that denotes the maximum number of units it can send (regardless the amount we order), which will arrive over from the current week up to L_1 weeks in the future according to an exogenous *arrival shares* process $(\rho_{t,0}^{J,i}, ..., \rho_{t,L_1}^{J,i})$ where $\sum_l \rho_{t,l}^{J,i} = 1$ and $\rho_{t,l}^{J,i} \geq 0$ for all i, t and l. The arrival quantity at lead time j from order $q_t^{J,i}$ can be denote as $o_{t,j}^{J,i} := \min(U_t^{J,i}, \tilde{q}_t^{J,i})\rho_{t,j}^{J,i}$. The LLT arrival quantities are defined similarly and denoted as $o_{t,j}^{L,i} := \min(U_t^{L,i}, \tilde{q}_t^{L,i})\rho_{t,j}^{L,i}$.

In brief, the overall sourcing processes from the initial order quantity to final arrivals can be modeled as

$$o_{t,j}^{J,i} := \min(U_t^{J,i}, f_p(q_t^{J,i}, \boldsymbol{M}_t^{J,i}))\rho_{t,j}^{J,i}, \quad (4)$$

$$o_{t,j}^{L,i} := \min(U_t^{L,i}, f_p(q_t^{L,i}, \boldsymbol{M}_t^{L,i}))\rho_{t,j}^{L,i}. \quad (5)$$

Internal Inventory Dynamics I_{t-}^i and I_t^i denote the on-hand inventory for product i at the beginning and end of a period t, respectively. The inventory update rule is given as follows,

$$I_{t-}^i = I_{t-1}^i + \sum_{j=0}^{L_1} o_{t,j}^{J,i} + \sum_{j=0}^{L_2} o_{t,j}^{L,i}, \quad (6)$$

and

$$I_t^i = \max\{I_{t-}^i - d_t^i, 0\}. \quad (7)$$

Reward Function We formulate the reward realized at t taking into account the current period inventory costs and sales. The construction of reward in our problem is such that it measures the periodic cash outflows caused due to replenishment and holding costs of inventory, and inflows are attributed to customer sales. It is defined as

$$R_t^i \triangleq p_t^i \min(d_t^i, I_{t-}^i) - c_t^{J,i} \sum_{j=0}^{L_1} o_{t,j}^{J,i} - c_t^{L,i} \sum_{j=0}^{L_1} o_{t,j}^{L,i} - h_t^i I_t^i. \quad (8)$$

Hence, computation of reward is essentially a function of history vector H_t and policy parameters θ, $R(\mathcal{H}_t, s_t^i, \theta)$.[1]

[1]Reward is a function of current period action a_t^i which is

2.2. The Dual Sourcing Optimization Problem

In our setting, we aim to maximize the total discounted [2] reward across the T length time horizon in expectation, while accounting for other constraints.

In the following, we state the dual sourcing optimization problem $\mathcal{P}_\mathbf{1}$,

$$\mathcal{P}_\mathbf{1}: \max_{\theta\in\Theta} \mathbb{E}\Big[\sum_{i\in\mathbb{A}}\sum_{t=0}^{T-1}\gamma^t R_t^i(\theta)\Big], \tag{9}$$

s.t.

$$I_0^i = \overline{I}^i, \tag{10}$$

$$\text{Equations}(3-7), \tag{11}$$

where Eq. (9) is the expression for initial inventory.

2.3. Scaling the Learning by Forecasting Sourcing Processes

In practice, we do not actually observe the full supply and arrival share processes for either the JIT or LLT sources. Under the IDP model described in the previous section, we only observe the arrivals share processes when an order was placed historically and we only observe the supply process when we do not receive the full order quantity. To handle this censoring we directly forecast the *arrivals* instead of the supply and arrival shares processes.

To see why this makes sense, note that the dynamics (6) and reward function (8) depend only on the arrivals $o_{t,j}^{J,i} := \min(U_t^{J,i}, f_p(q_t^{J,i}, \boldsymbol{M}_t^{J,i}))\rho_{t,j}^{J,i}$ and $o_{t,j}^{L,i} := \min(U_t^{L,i}, f_p(q_t^{L,i}, \boldsymbol{M}_t^{L,i}))\rho_{t,j}^{L,i}$. Thus, for the purposes of constructing our simulator from historic data, we forecast arrivals conditional on the action a_t^i rather than estimating the post-processing behavior and the supply and arrival share processes.

Arrivals For arrivals, denoting by $H_{t,O}^i$ the observed components of H_t^i, one can forecast

$$p(o_{t,0}^{J,i}, \ldots, o_{t,L_1}^{J,i} | H_{t,O}^i; \psi_1). \tag{12}$$

Note that conditioning on $H_{t,O}^i$, θ is equivalent to conditioning on $H_{t,O}^i$, ψ_1 and the past JIT order actions a_s^i for $s \le t$.

3. Experimental Evaluations

3.1. Training/Evaluation Configurations

Real-world Dataset We use the same real-world dataset as used in (Madeka et al., 2022) with approximately 80,000 products for 124 weeks from April 2017 to August 2019. Out of the 124 weeks, we treat the first 72 weeks as training dataset and the remaining 52 as the backtesting dataset. However, in future iterations, we plan to train the DRL models on 104 weeks so that the policy agents can better track seasonal patterns.

Baselines We compare our Dual Sourcing RL (`DualSrc-RL`) buy policy against several baselines, i.e. `BaseStockHorizonTip (BSHT)`, *Tailored Base Surge (TBS)*, *Just-in-time RL (JIT-RL)*. The first two are classic operation research baselines and the *JIT-RL* is the RL baseline where the RL model is trained as a single-sourcing model (Madeka et al., 2022). More detailed description of those baselines are reported in Appendix A.1.

3.2. Main Results and Analytics

We use the training dataset to train the RL algorithms and perform evaluation experiments for the compared algorithms over the test dataset. The evaluation results are reported in Table 1.

Setting	Method	% of `BSHT`
JIT Policy	`BSHT`	100
	`JIT-RL`	104.78
Dual Sourcing	`TBS`	117.69
	`DualSrc-RL`	121.54

Table 1: Cumulative discounted rewards (as % of `BSHT`) for 52 backtest periods for different policy methods.

From the results above, we can observe that overall dual sourcing strategies `DualSrc-RL`, `TBS` are favorable in terms of rewards. Furthermore, `DualSrc-RL` is most profitable in all the run scenarios, and has 4% reward gains over `TBS`.

4. Capacity Management with Neural Coordination

Having formulated the unconstrained inventory control problem in $\mathcal{P}_\mathbf{1}$, now we consider a constrained situation, where network capacity constraints are introduced, and considered as part of the exogenous process. We are interested in studying the constrained problem because it is a challenging variant in real-world supply chain applications. A large retailer typically manages a supply chain for multiple products and has limited resources (such as storage) that are shared amongst all the products that retailer stocks. Specifically, we have the following set of formulas representing the storage capacity constraints,

$$G := \{K_0, K_2, ..., K_{T-1}\}. \tag{13}$$

computed via policy π parameterized by θ, and input $\mathcal{H}_t$.

[2] This discounted reward implies time value of actual cash flows. The discounting factor here is γ.

The constrained problem $\mathcal{P}_\mathbf{1}$ can be obtained by adding the capacity constraints (15) to $\mathcal{P}_\mathbf{1}$,

$$\mathcal{P}_\mathbf{2}: \max_{\theta\in\Theta} \mathbb{E}\Big[\sum_{i\in\mathbb{A}}\sum_{t=0}^{T-1}\gamma^t R_t^i(\theta)\Big], \tag{14}$$

s.t.

$$\sum_{i\in A} v^i I_t^i \le K_t, . \tag{15}$$

For the reward function, we modify the unconstrained reward to incorporate a penalty according to the capacity prices,

$$R_t^{\lambda,i} \triangleq R_t^i - \lambda_t v^i I_t^i, \tag{16}$$

where λ_t is the storage capacity price at time t.

4.1. Forecasting the Capacity Prices by a Neural Coordinatator

The role of coordinator agent is to predict capacity prices for the targeting inventory network given any capacity constraints as input. One example of a coordination mechanism is model predictive control (MPC), which would use forecasted demand to perform a dual cost search for the next L periods. However, for an RL buying policy that uses many historical features, model predictive control would require forward simulating many features and it may be difficult to model the full joint distribution of all these features.

Instead, we apply a coordination approach as used in (Eisenach et al., 2024) by introducing a deep learning model for this problem. Specifically, we train a neural network to forecast the future prices of capacity that would be required to constrain the dual sourcing RL buying policy. Specifically, we learn a neural network to predict,

$$(\lambda_t, \lambda_{t+1}, ..., \lambda_{t+L}) = \phi_\omega(H_t, G), \tag{17}$$

where H_t denotes the historical state vector and G denotes the capacity constraints.

The training algorithm for the neural coordinator can be found in Appendix.

4.2. Evaluation Results

In this section, we backtest our proposed dual sourcing buy policy with neural coordinator for storage capacity management.

Compared Policies and Baselines We compare our unconstrained Dual Sourcing RL (`DualSrc-RL`) buy policy (as the baseline) with its two constrained variants. For the first variant, the neural coordinator is used to constrain the `DualSrc-RL` agent, and a MPC predictor is used to constrain `DualSrc-RL` as the second variant.

Performance Metrics In addition to measuring reward achieved by the various policies, we consider several additional measures of constraint violation. They are (M1) mean constraint violation (M2) percent of weeks where the violation exceeded 10%.

Results The Table 2 shows the results on the out-of-training backtesting period, where each combination of policy and coordinator were evaluated against 100 sampled storage constraint paths. Note that under all metrics (both violation and reward) the `DualSrc-RL` policy with neural coordinator outperforms `DualSrc-RL` with MPC. Although some of the violation metrics seem somewhat high, one should keep in mind that many of the capacity curves sampled will be highly constraining, much more so than in a real-world setting – in practice if the supply chain were so constrained, one would build more capacity.

Table 2: Out-of-distribution evaluation results.

		Violations		
Buy Algo	Coordinator	M1	M2	Reward
DualSrc-RL	-	28.9%	35.8%	**100**
DualSrc-RL	Neural	**3.5 %**	**9.1%**	99.7
DualSrc-RL	MPC	13.3%	18.2%	96.9

5. Conclusion and Future Work

In this paper, we investigates ML-based inventory control methodologies with the consideration of stochastic supply chain processes that introduce scalable representation models to mitigate the supply risk and optimize the complex cross-product constraints/resources. Specifically, we investigate how to efficiently build a Deep RL framework for the problem by forecasting the real-world supply chain processes under a range of assumptions. We highlight that instead of directly modeling the complex physical constraints into the learning pipeline and solving the problem as a whole, our approach breaks down those supply chain processes into different DL modules, leading to improved performance on larger real-world retail datasets. For future research, it is interesting to develop efficient modules for approximating other cross-product dependencies in real-world supply chain networks, e.g. containerization processes in global transportation, truck-load and placement.

References

Balaji, B., Bell-Masterson, J., Bilgin, E., Damianou, A., Garcia, P. M., Jain, A., Luo, R., Maggiar, A., Narayanaswamy, B., and Ye, C. Orl: Reinforcement learning benchmarks for online stochastic optimization problems, 2019.

Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

Eisenach, C., Ghai, U., Madeka, D., Torkkola, K., Foster, D., and Kakade, S. Neural coordination and capacity control for inventory management, 2024. URL https://arxiv.org/abs/2410.02817.

Fukuda, Y. Optimal policies for the inventory problem with negotiable leadtime. *Management Science*, 10(4): 690–708, 1964.

Madeka, D., Torkkola, K., Eisenach, C., Luo, A., Foster, D., and Kakade, S. Deep inventory management, 2022.

Maggiar, A., Dicker, L., and Mahoney, M. Consensus planning with primal, dual, and proximal agents. *arXiv preprint arXiv:2408.16462*, 2024.

Whittemore, A. S. and Saunders, S. Optimal inventory under stochastic demand with two supply options. *SIAM Journal on Applied Mathematics*, 32(2):293–305, 1977.

Xin, L. and Goldberg, D. A. Asymptotic optimality of tailored base-surge policies in dual-sourcing inventory systems. *Management Science*, 64(1):437–452, 2018.

A. Related work

Dual Sourcing Inventory Control: Dual sourcing is a heavily studied problem in inventory control literature, and has become a common practice supply chain organizations worldwide. In vanilla dual sourcing problem, the sourcing channels i.e., JIT, LLT essentially cause a trade-off in lead time vs ordering costs since usually LLT (direct import sourcing channel) has lower sourcing cost. In this context, Tailored Base-Surge (TBS) policy is a widely used method, wherein at each period, a constant order is placed via the LLT channel, and a dynamic order is placed to match an inventory order-up-to level via the JIT channel respectively. The JIT channel's order-up-to level essentially implies a *Safety Stock* maintained to address sudden *demand surges* (Xin & Goldberg, 2018). Although, such policies present fairly intuitive approaches to handle the dual sourcing problem, but the optimality analysis of such policies has been a hard problem in general (Whittemore & Saunders, 1977), except for certain edge cases. Interestingly, the optimal policy is shown to be vanilla base-stock policy when the lead time difference between LLT and JIT channels is exactly 1 period (Fukuda, 1964).

Reinforcement Learning for Inventory Control: Recently, application of Reinforcement Learning (RL) methods to produce ordering decisions in large scale inventory management systems has gained significant attention. In this context, Deep RL approaches have emerged more recently over its other *Model-Based* counterparts, due to improved computational scalability and generalization performance of Deep Neural Network (DNN) architectures. Furthermore, DRL methods have been shown to achieve performance gains over benchmarks for the multi-period newsvendor problem under fairly realistic assumptions on costs, prices, demand and stationarity (Balaji et al., 2019). It is worth highlighting that DRL for single sourcing problem has been comprehensively investigated with extensive empirical evidences at Amazon (Madeka et al., 2022).

Capacity Management and Coordinatation Mechanism Retailers typically manage a supply chain for multiple products and has limited capacity resources (such as storage) that are shared amongst all the products that retailer stocks. The classic method for handling capacity constraints is to call a simulation and optimization process to compute shadow prices on the shared resources. Model predictive control (MPC) consists of using a model to forward simulate a system to optimize control inputs and satisfy any constraints. At each time step, one re-plans based on updated information that has become available in order to select the next control input. Recent work Maggiar et al. (2024) introduced the Consensus Planning Protocol (CPP), which targets problems where multiple agents (each of which is locally optimizing its own utility function) all consume a shared resource. This is closely related to a distributed ADMM procedure (Boyd et al., 2011). Another work (Eisenach et al., 2024) presented a new capacity control mechanism for RL-based buying policies and proposed a Neural Coordinator model to generate forecasts of capacity prices. Their formulation of capacitated inventory management can be viewed as a special case of CPP (a central coordinator adjusts prices, and the other agents adjust their plans).

A.1. Baseline Methods

A.1.1. Improved Tailored Base Surge Policy

We adopt a modified version of vanilla Tailored Base Surge (TBS) policy which has been described in (Xin & Goldberg, 2018). For each product i, TBS policy will place a *dynamic* LLT order $q_{L,t}^{i,\texttt{TBS}}$ every period. In our problem setting, this LLT order will arrive with lead-time of δ_L.

Whereas, for orders through JIT channel, this TBS policy will first calculate a dynamic target order-upto-level $I_t^{i,\texttt{Tip}}$ via the production "Horizon Tip Calculator" method. Consequently, TBS policy places also places a dynamic order for the JIT channel i.e., $q_{J,t}^{i,\texttt{TBS}}$ that can bring back the inventory level to $I_t^{i,\texttt{Tip}}$. With on-hand inventory at the end of the period, I_t^i, the JIT order at time t is given by:

$$q_{J,t}^{i,\texttt{TBS}} = \max\left\{0, I_t^{i,\texttt{Tip}} - I_t^i - \left[\sum_{\tilde{t}=0}^{t-1}\sum_{k=t}^{t+\delta_t^{i,\texttt{Pred}}} \left(o_{\tilde{t},k-\tilde{t}}^{J,i} + o_{\tilde{t},k-\tilde{t}}^{L,i}\right)\right]\right\}, \tag{18}$$

where $\delta_t^{i,\texttt{Pred}}$ is the median forecasted VLT at time t for product i. In other words, Improved TBS subtracts on-hand and inflight inventory from the Horizon Tip to compute JIT orders for current period. The order quantities $q_{J,t}^{i,\texttt{TBS}}, q_{L,t}^{i,\texttt{TBS}}$ will arrive according to the underlying quantity over time arrival models.

Choice of LLT Order Input: We set LLT order quantities $q_L^{i,\texttt{TBS}}$ as scaled 12-week rolling mean of product-level demands from training set, i.e., $q_{L,t}^{i,\texttt{TBS}} = \alpha \cdot \frac{1}{12} \cdot \sum_{\tilde{t}=t-11}^{t-1} d_{\tilde{t}}^i$. The order scaling factor α is used in our experiments as a search parameter for getting optimal TBS policy.

A.1.2. OTHER BASELINE POLICIES

1. We use a single source JIT Base Stock Policy with orders dictated by eq. (18) which we call `BaseStockHorizonTip` (BSHT).
2. The existing RL buying policy for JIT single source from the literature (Madeka et al., 2022).

B. Training Algorithms

B.1. Training Algorithm for the Buy Policy

Firstly, recall that parameters $\theta \in \Theta$ at any time t essentially dictates the RL agent's policy as expressed in eq. (3), therefore our problem reduces to learning optimal parameters $\theta^\star \in \Theta$. Specifically, we leverage a Deep Neural Network (DNN) architecture for $\pi(\cdot, \cdot; \theta)$. So, θ in our DRL framework essentially implies the weights and parameters of the constituent neurons in the policy network.

Observe that constraints of $\mathcal{P}_\mathbf{1}$ (9) are in fact definitions for different components of the reward function, and, therefore can be omitted from the optimization problem formulation otherwise. Next, we present the Direct Backpropagation (`DirectBP-DualSrc`) training algorithm for DRL policy.

Algorithm 1 Direct Backpropagation DRL training algorithm (`DualSrc-RL`)

Input: set of products $\mathcal{A}$, training batch size M, step size: η, $\theta_0 \in \Theta$.
Initialize: Batch Iterator: $b \leftarrow 1$.
while θ is not converged **do**
  Sample mini-batch of products $\mathcal{A}_b$ of size M from $\mathcal{A}$. and set $R^b \leftarrow 0$.
  for $i \in \mathcal{A}_b$ **do**
    $R^i \leftarrow 0$, $I_0^i \leftarrow \overline{I}^i$.
    for $t = 0, \ldots, T^{\texttt{Train}} - 1$ **do**
      Place orders $\left(q_{J,t}^i, q_{L,t}^i\right) = \pi_t^i(H_t; \theta_{b-1})$.
      Sample JIT, LLT arrivals $(o_{t,0}^{J,i}, \ldots, o_{t,L_1}^{J,i}, o_{t,0}^{L,i}, \ldots, o_{t,L_2}^{L,i})$.
      Update inventory I_t^i according to (6) and (7).
      Collect reward R_t^i according to (8)
      $R^i \leftarrow R_t^i + \gamma^t \cdot R_t^i$.
    end for
    $R^b \leftarrow R^b + R^i$.
  end for
  $\theta_b \leftarrow \theta_{b-1} + \eta \cdot \nabla_\theta \mathcal{P}_1^b\big|_{\theta=\theta_{b-1}}$. *// Update Parameters of Policy Network* $\pi(\cdot, \cdot; \theta)$.
  $b \leftarrow b + 1$.
end while=0

B.2. Training Algorithm for the Neural Coordinator

In the next, we introduce the optimization problem for learning the neural coordinator. First, assume a fixed dual sourcing buying policy θ. Below we describe the ways in which the coordinator's Exo-IDP deviates from the buying agent's Exo-IDP.

The coordinator solves the following problem:

$$\mathcal{P}_\mathbf{3} : \min_{\omega \in \Omega} \mathbb{E}\Big[\sum_{t=0}^{T-1} (\sum_{i \in A} v^i I_t^i - K_t)_+^2 + ||\lambda_t|| + \mathcal{L}\left(\lambda_t, H_t^\lambda\right) \Big], \tag{19}$$

$$\tag{20}$$

where $\mathcal{L}\left(\lambda_t, H_t^\lambda\right)$ denotes the total capacity price forecast error at time t,

$$\mathcal{L}\left(\lambda_t, H_t^\lambda\right) = \sum_{s=1}^{L} ||\lambda_t - (\hat{\lambda}_{t-s}^L)_{L-s}||^2, \tag{21}$$

which is the mean squared error (MSE) of all past forecasts for the current cost.

We implement of the training algorithm for learning the neural coordinator for our dual sourcing problem. Specifically, we train the coordinator by solving the problem ($\mathcal{P}_3$). Similar to the training algorithm for the dual sourcing buy policy, we apply the Direct Backpropagation algorithm to optimize the loss objective 19 in ($\mathcal{P}_3$). The pseudo code of the training algorithm is shown in Algorithm 2.

Algorithm 2 Training algorithm for the Neural Coordinator

Input: set of products $\mathbb{A}$, training batch size M, step size: η, given buy policy θ, $\{\overline{I}^i\}_{i\in\mathbb{A}}$, δ_L, ϵ, initial neural coordinator $\omega_0 \in \Omega$.
Initialize: Batch Iterator: $b \leftarrow 1$.
while stop criterion is not satisfied **do**
  Sample mini-batch of products $\mathbb{A}_b$ of size M from $\mathbb{A}$. and set $R^b \leftarrow 0$.
  for $i \in \mathbb{A}_b$ **do**
    $R^i \leftarrow 0$, $I_0^i \leftarrow \overline{I}^i$.
    for $t = 0, \ldots, T^{\texttt{Train}} - 1$ **do**
      Collect instantaneous state and history vector $s_t^i, \mathcal{H}_t$.
      Place orders $\left(q_{J,t}^i, q_{L,t}^i\right) = \pi_t^i(\mathcal{H}_t, s_t^i; \theta)$.
      Sample JIT, LLT arrivals $(o_{t,0}^{J,i}, \ldots, o_{t,L_1}^{J,i}, o_{t,0}^{L,i}, \ldots, o_{t,L_2}^{L,i})$.
      Collect reward R_t^i and update inventory I_t^i.
    end for
  end for
  Update global state and compute coordination loss by Equation 19.
  $\omega_b \leftarrow \omega_{b-1} + \eta \cdot \nabla_\omega \mathcal{P}_3^b\big|_{\omega=\omega_{b-1}}$. *// Update Parameters of the Neural Coordinator* $\phi(\cdot, \cdot; \omega)$.
  $b \leftarrow b + 1$.
end while=0

C. Input Features

C.1. Featurization for Buying Policy

In terms of features, we mainly use the following feature list provided to the buying policy:

1. The current inventory level
2. Previous actions aiu that have been taken
3. Time series features
   (a) Historical availability corrected demand
   (b) Distance to public holidays
   (c) Historical website glance views data
4. Static product features
   (a) Product group
   (b) Text-based features from the product description
   (c) Brand
   (d) Volume
5. Economics of the product - (price, cost etc.)
6. Capacity costs – past costs and forecasted future costs

C.2. Featurization for Neural Coordinator

The neural coordinator takes the following aggregate/population level features:

1. Aggregated actions, inventory, demands for all current and previous times
    (a) Order quantities
    (b) Inventory
    (c) Availability corrected demand
    (d) Inbound
    (e) All the above, but weighted by inbound and storage volumes
2. Forecasted quantities for next L weeks.
    (a) Mean demand at lead time
    (b) Inventory after expected drain at lead time
    (c) All the above, but weighted by inbound and storage volumes
3. Other time series features
    (a) Distance to public holidays
    (b) Mean economics of products - (price, cost etc.), weighted by demand and volumes
4. Capacity costs (past costs and forecasted future costs)

D. Complementary Results of the Neural Coordinator in Addition to Sec. 4.2

The Figure 1 below demonstrates two examples of trajectories in the evaluation period for our `DualSrc-RL` (DS-RL) buy policies with their coordinator settings (unconstrained, Neural coordinator or MPC). We can see that the neural coordinator is able to constrain the on hand inventory within the capacity limit on the out-of-training backtesting period.

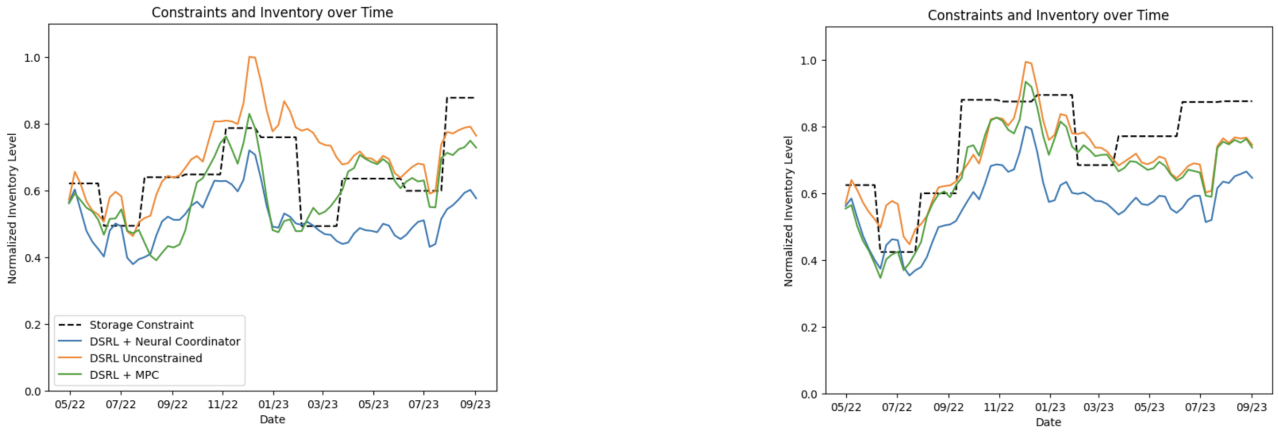

Figure 1: Inventory trajectories under different constraint paths during the out-of-training period.