Engineering Applications of Artificial Intelligence 134 (2024) 108670

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Neuroevolution reinforcement learning for multi-echelon inventory optimization with delivery options and uncertain discount

Zakka Ugih Rizqi [a,*], Shuo-Yan Chou [a,b]

[a] Department of Industrial Management, National Taiwan University of Science and Technology, No. 43, Keelung Rd., Taipei, 10607, Taiwan
[b] Intelligent Manufacturing Innovation Center, National Taiwan University of Science and Technology, No. 43, Keelung Rd., Taipei, 10607, Taiwan

ARTICLE INFO

Keywords:
Multi-echelon inventory
Optimization
Reinforcement learning
Simulation
Supply chain

ABSTRACT

The advanced information technology has enabled supply chain to make centralized optimal decision, allowing to make a global optimal solution. However, dealing with uncertainty is important in inventory management. Besides demand and supply uncertainties, supplier discounts also often arise unexpectedly. Further, suppliers or third-parties typically offer various delivery options in which trade-off occurs between cost and lead time. Thus, this study introduces new problem namely Multi-Echelon Inventory Optimization with Delivery Options and Uncertain Discount (MEIO-DO-UD). As a solution, Neuroevolution Reinforcement Learning (NERL) framework is developed for minimizing total system cost. The environment is modeled via System Dynamics (SD) and actor is presented by integration of Artificial Neural Network and Evolutionary Algorithm (EA), creating an effective decision-making model under dynamic uncertainty. The experimental study has been conducted where two different supply chain networks are given namely serial and divergence. Three EA algorithms are compared namely Differential Evolution (DE), Memetic Algorithm (MA), and Evolution Strategy (ES). Furthermore, NERL is also compared with the EA-optimized classical continuous review model namely (s,Q). The result shows that regardless what EA type is used, the proposed NERL always outperforms EA-optimized (s,Q) model. The more complex the problem, the further improvement can be made i.e. cost reduction up to 58%, followed by the fill rate improvement. The result also shows that NERL can avoid overfitting. Managerial implications are highlighted where NERL provides the more stable inventory level among all supply chain partners and bull-whip effect can be damped.

1. Introduction

Due to high uncertainty environment in supply chain, inventory is hold to handle the fluctuation of demand and supply (Baruah et al., 2016). While having more inventory can enhance service levels, it also results in elevated costs particularly holding cost. Thus, effective inventory management is essential to strike the right balance between service levels and inventory expenditures (Gallego-garcía et al., 2021). It is worth noting that inventory typically represents a substantial portion, ranging from 20% to 60%, of a manufacturing company's total assets (Anyibuofu, 2014). This indicates the importance of effective inventory management.

Many methods are employed in inventory management for shaping the profitability of the businesses. It was started with the well-known mathematical model so-called Economic Order Quantity (EOQ) for determining the minimum cost for lot sizing (Harris, 1990). The development was then continued by managing inventory through inventory control system such as two classical inventory models namely continuous review (Mahapatra et al., 2021) and periodic review (Zhu et al., 2022). Furthermore, due to development of information technology, inventory decision is often integrated with production planning such as Material Requirements Planning (MRP), Manufacturing Resources Planning (MRP II), Advanced Planning and Scheduling (APS), and Enterprise Resources Planning (ERP) (Shofa and Widyarto, 2017) (Chandraju et al., 2012). However, most of proposed methods work under strict assumption namely deterministic and static condition. The study conducted in 2011 illustrates that using those assumptions can prove to be quite expensive when applying the corresponding models to real-life situations (Tunc et al., 2011). Even more, most methods only consider single-echelon model in which does not consider the other parties leading to local optimal solution. When mathematical model is used for multi-echelon inventory management, the problem can quickly become too complex and computationally expensive due to growing

* Corresponding author.
E-mail address: ugihzakka@gmail.com (Z.U. Rizqi).

https://doi.org/10.1016/j.engappai.2024.108670
Received 13 January 2024; Received in revised form 18 April 2024; Accepted 21 May 2024
Available online 28 May 2024
0952-1976/© 2024 Elsevier Ltd. All rights are reserved, including those for text and data mining, AI training, and similar technologies.

Abbreviations

NERL	Neuroevolution Reinforcement Learning
FR	Average Fill Rate
TC	Total Cost

Sets

I	Set of facilities, $I = \{i \mid i \in I\}$
T	Set of time, $T = \{t \mid t \in T\}$
R	Set of retailers, $R = \{r \mid r \in R\}$
S	Set of suppliers, $S = \{s \mid s \in S\}$
D	Set of delivery options, $D = \{d \mid d \in D\}$

Parameters

$OC(i,t)$	Ordering cost of facility i at t
ρ	Holding cost per unit per t
γ	Shortage cost per unit
$v(d)$	Variable cost per unit purchased
$A(d)$	Transportation cost per delivery
$C(d)$	Maximum capacity for each transportation
$ST(r,t)$	Number of shortages in retailer r at t
$\mathscr{D}(r,t)$	Number of demands in retailer r at t
$\mathscr{S}(i,t)$	Number of supplies arrived in facility i at t
$D(i,t)$	Number of demands in facility i at t
$IC(i)$	Inventory capacity in facility i
$IH(i,t)$	Inventory on hand in facility i at t
M	Large number as penalty cost

States

$IP(i,t)$	Inventory position of facility i at t
$DC(s,t)$	Discount given for variable cost by supplier s at t

Actions/Variables

$Q(i,t)$	Order quantity of facility i at t {Integer}
$DO(d,i,t)$	Delivery option d is chosen in facility i at t {Binary}
$f(s,i,t)$	Supplier s supplies facility i at t {Binary}

decision variables. This complicates the connection between models and practical real-world applications.

Many previous studies have proposed the derived version of inventory model to consider uncertainty and dynamicity. A study (Patriarca et al., 2020) derived EOQ model to consider demand uncertainty and time-dependent product quality via Monte Carlo simulation. Using similar method, some other studies considered lead time (Rizqi and Khairunisa, 2020) and defective product (Rizqi et al., 2021) as uncertainty factors besides the demand by deriving Min-Max inventory method. Another idea to consider uncertainty in inventory is by utilizing predictive methods (Prak and Teunter, 2019) such as utilizing ensemble deep learning for order-up-to-level inventory optimization for demand forecasting (Seyedan et al., 2023). However, none of studies has considered not only supply and demand uncertainties, but also the discount event uncertainty given by supplier at a time. This discount event also can be seen as an uncertain or dynamic price which has become more common recently. Previous studies tried to comprehend the EOQ model with discount quantity but still in static and certain conditions only (Kristiyani and Daryanto, 2019) (Firoozi et al., 2013). Further, many previous studies also only assume single-delivery option in inventory decision or even not considering transportation at all when single-echelon inventory is considered. Traditional inventory management literature predominantly focuses on a single-sourcing model characterized by static customer demand and a deterministic lead time (Liu et al., 2022). In many multi-echelon practical situations, supplier or third-party logistics usually offer multiple delivery options or sometimes called as multi-sourcing or multi-mode with the trade-off in cost and responsiveness.

Therefore, considering all previous features, it leads to a new inventory problem namely Multi-Echelon Inventory Optimization with Delivery Options and Uncertain Discount (MEIO-DO-UD). The purpose of this study is to provide an effective model that can be practically useful by applying Neuroevolution Reinforcement Learning (NERL) for multi-period, multi-echelon, multi-network, and single-product inventory problems. Compared to the other reinforcement learning algorithms, NERL can avoid sample inefficiency without the need to update Q-table, better exploration, flexibility, and faster convergence. The systematic experiment is given to verify the effectiveness and the efficiency of the proposed model in solving MEIO-DO-UD under different setups of algorithms. Comparative analysis is also conducted to ensure the superiority of the proposed model compared to the classical continuous inventory review model i.e. (s,Q) including its impact on the bull-whip effect.

The remaining parts of this paper are structured as follows. In section 2, an examination of the literature review outcomes is provided, including a comprehensive overview of reinforcement learning for inventory problem, NERL, along with a comparison between this study to the previous studies. In section 3, the methodology is elaborated in detail, outlining the design of the reinforcement learning algorithm, simulation modeling, and how total cost is measured as expected reward. Section 4 entails the case study given where serial and divergence supply chain networks are used, including the result and discussion, measuring the statistical difference between the proposed method compared to other methods. Subsequently, section 5 furnishes conclusions aimed at summarizing the undertaken tasks and outcomes, while also proposing potential directions for future research endeavors.

2. Literature review

Inventory management has vital functions in business for both small scale i.e. operations management (Guo et al., 2019) and large scale i.e. supply chain management (Singh and Verma, 2018). The main objective in inventory management is to minimize total system cost while meeting customer service level. Inventory management often involves sequential decision-making challenges that can be represented as Markov Decision Processes (MDPs), making them suitable for the application of Reinforcement Learning (RL) (Wu et al., 2023) (Boute et al., 2022). In contrast to analytical-based models, RL views the problem as a simulated environment, thus avoiding oversimplification of reality and allowing for more flexibility, especially for considering a larger number of variables (Kosasih and Brintrup, 2022). RL constitutes one of the three primary machine learning paradigms, alongside supervised and unsupervised learning. RL involves agent or actor, which essentially act as domain experts, making decisions and taking actions under specific states (Almahamid and Grolinger, 2021). Unlike supervised learning, RL does not rely on labeled data to learn, instead, it acquires knowledge through experiential learning by interacting with its environment, observing outcomes, and adapting its behavior accordingly as illustrated in Fig. 1. In general reinforcement learning algorithms, the main objective is to find an optimal policy ($X^{\pi}(S_t)$) based on certain state (S) in time t in order to maximize the expected reward as shown in Equation (1). This might include uncertain information in the initial state S_0, thus, the expected operator is used to average among random variables.

$$\max_{\pi} \mathbb{E}\left\{ \sum_{t=0}^{T} C(S_t, X^{\pi}(S_t)) | S_0 \right\} \tag{1}$$

There are a lot of RL algorithms developed such as Q-learning and Proximal Policy Optimization (PPO). In a complex problem, neural

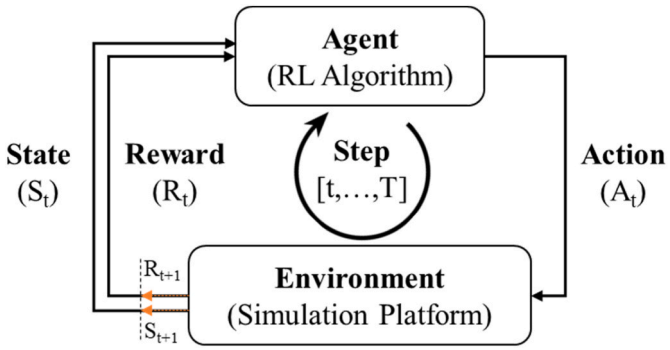


Fig. 1. A general architecture of reinforcement learning.

network is used as function approximation to map actions from states that leads to the growth of Deep Reinforcement Learning (DRL) (Wang et al., 2022). Despite of classical reinforcement learning algorithms, NERL has attracted a lot of attention due to its competitiveness in many applications such as nuclear energy (Radaideh et al., 2023), games (Barmi et al., 2011), and self-driving cars (AbuZekry, 2019). It essentially involves applying evolutionary algorithms to create and train neural networks (Heidrich-Meisner and Igel, 2009). Evolutionary algorithms (EAs) operate on a straightforward principle, mirroring the mechanisms of natural selection and evolution (Slowik and Kwasnicka, 2020). Some well-known algorithms in EA that have been successfully used to optimize neural network are Differential Evolution (DE) (Baioletti et al., 2020), Memetic Algorithm (MA) (García-Ródenas et al., 2021), and Evolution Strategy (ES) (Friedrich and Maziero, 2023).

NERL lies in the direct policy search with gradient-free process. NERL also belongs to the DRL once the neural network has many hidden layers. It has advantages ranging from better exploration to the ease of parallelization and potentially more robust to the hyper-parameters selection (Zhang et al., 2022). Compared to the other RL algorithms such as Deep Q Network (DQN) and PPO, they face some disadvantages, especially for solving MEIO. First, DQN and PPO algorithms work iteratively for each time step that needs to collect data and make an update such as the Q-Table by using Bellman's equation. This will lead to sample inefficiency and slower convergence. Second, DQN faces difficulty with a large number of discrete actions. Even though PPO is more flexible, PPO can still face challenges in learning effective representations. Third, both algorithms have limited exploration since DQN relies on epsilon-greedy strategies while PPO uses policy gradients for exploration. These disadvantages can be solved through NERL due to the capability of metaheuristics. In addition, this study also designs NERL as single-agent instead of multi-agent e.g. multi-agent deep reinforcement learning algorithm (Zhou et al., 2024) since it has better computational complexity for large-scale problems.

Inventory model has developed rapidly in literature following the more complex condition in reality. Due to effectivity, data-driven framework has been used increasingly by emphasizing on optimality (Rekabi et al., 2023) or practicality (Geevers et al., 2022), from single-echelon inventory (Kara and Dogan, 2018) to multi-echelon inventory (Rizqi, 2023). Nevertheless, many previous studies in proposing inventory models under uncertainty only consider demand variability with constant lead time (Dittrich and Fohlmeister, 2021), (Prestwich et al., 2012), (Kaynov et al., 2024) avoiding the stochasticity of the lead time even though it is practical (Geevers et al., 2022). In terms of decision-making, most studies also only consider providing order quantity (Kaynov et al., 2024) (Peng et al., 2019). Whereas, multiple decisions may be needed such as delivery options where several modes are available offering the trade-off between delivery time and cost. By seeing this gap, this study takes one step further in inventory management by providing multi-echelon inventory optimization model considering not only demand and lead time uncertainties but also discount uncertainty with delivery options decision, in addition to order quantity. To achieve the minimum expected total cost, NERL framework is proposed by utilizing the integration of evolutionary algorithm as the policy optimizer and neural network as function approximator in mapping state to action. In other words, evolutionary algorithm will help to optimize the weight and bias of neural network to reach near-global optimal solution. In previous studies, Q-Learning and Deep Q-Learning were used (Oroojlooyjadid et al., 2022). However, they may lead to other issues where the action space becomes larger and back propagation cannot provide satisfactory results.

Several relevant studies that solve multi-echelon inventory problem with reinforcement learning are reviewed and compared with this study as shown in Table 1. It can be seen that there are already a few studies utilizing different RL methods but only one study that have utilized NERL based on Evolution Strategy without any capacity constraint and complicated uncertainties considered. Further, they result show the superiority of NERL algorithm compared to the stochastic programming for small and large scale of simple inventory problem. In terms of application to inventory management, most of studies considers only uncertain demand while assuming other factors as certain or deterministic. In terms of supply chain network, some studies have considered serial and divergence network. Most of studies also do not consider delivery options. Different types in handling unmet demand are classified into three groups namely backorders, lost sales, and hybrid or so-called as partial backorder where some customers are willing to wait through backorder but others are not. It can be seen that this study differs with the others by emphasizing not only on the methodology, but also introducing the comprehensive and practical problems in multi-echelon inventory problem so-called MEIO-DO-UD.

3. NERL for multi-echelon inventory optimization

Since the ultimate goal of this study is to solve a complex multi-echelon inventory problem with multiple uncertain factors and deliver options, calculating the exact optimal policy is intractable. Thus, NERL with System Dynamics (SD) model is used as the main framework for performing dynamic optimization. The proposed NERL at the top level is shown in Fig. 2 while the more detailed is presented in the form of pseudocode as presented in Table 2. Fig. 2 visualizes that there are basically 2 main components as general reinforcement learning framework has namely agent and environment. Both components work interactively where environment provides the inventory-related states via SD simulation and agent provides the optimal actions via ANN prediction tuned by EA algorithm depending on the states given each time step. The actions given will be simulated again by SD which influences the next states. At the same time, SD returns the reward. In the end of simulation time, the reward is cumulated and the first episode is finished. This process is repeated until the certain number of episodes independently to measure the expected cumulative reward. The following subsection describes each component in detail.

3.1. Agent development based on NERL algorithm

The agent is built based on the integration of Evolutionary Algorithm (EA) and Artificial Neural Network (ANN) so-called NERL. EA is used to optimize the weights and biases of ANN. Thus, the chromosome length of EA depends on the ANN topology as indicated at agent box in Fig. 2. In the context of dynamic optimization, integration of EA and ANN is needed since general ANN that is generally optimized by backpropagation needs historical data for training as in supervised learning. Whereas, in reinforcement learning, the data will be generated dynamically that strongly depends on the action given. By replacing backpropagation by EA, the requirement of historical data can be avoided.

The choice of what EA should be coped with ANN is also quite interesting to be investigated. In this study, since the objective of EAs is

Table 1
Summary of relevant studies in using RL for multi-echelon inventory.

Study	Approach	Objective	Multi-Echelon		Uncertainty Factors			Delivery Options	Unmet Demand
			Serial	Divergence	Lead Time	Demand	Discount		
Dittrich and Fohlmeister (2021)	DQN	Min. Cost	✓	–	–	✓	–	–	Backorders
Liu et al. (2022)	HAPPO	Min. Cost	✓	✓	–	✓	–	–	Hybrid
Geevers et al. (2022)	PPO	Min. Cost	✓	✓	✓	✓	–	–	Backorders
Prestwich et al. (2012)	NERL	Min. Cost	✓	✓	–	✓	–	–	Lost sales
Kosasih and Brintrup (2022)	MA-A2C	Min. Cost	✓	–	–	✓	–	–	Lost sales
Gijsbrechts et al. (2022)	A3C	Min. Cost	–	✓	–	✓	–	✓	Lost sales
Oroojlooyjadid et al. (2022)	DQN	Min. Cost	✓	–	–	✓	–	–	Backorders
Kaynov et al. (2024)	PPO	Min. Cost	–	✓	–	✓	–	–	Hybrid
Peng et al. (2019)	VPG	Max. Profit	–	✓	–	✓	–	–	Backorders
Zarandi et al. (2013)	Fuzzy-RL	Min. Cost	–	✓	–	✓	–	–	Backorders
Ours	SD-NERL	Min. Cost	✓	✓	✓	✓	✓	✓	Lost sales

*PPO: Proximal Policy Optimization; DQN: Deep Q-Learning; HAPPO: Heterogeneous Agent Proximal Policy Optimization; MA-A2C: Multi-Agent Advantage Actor-Critic; A3C: Asynchronous Advantage Actor Critic; VPG: Vanilla Policy Gradient.

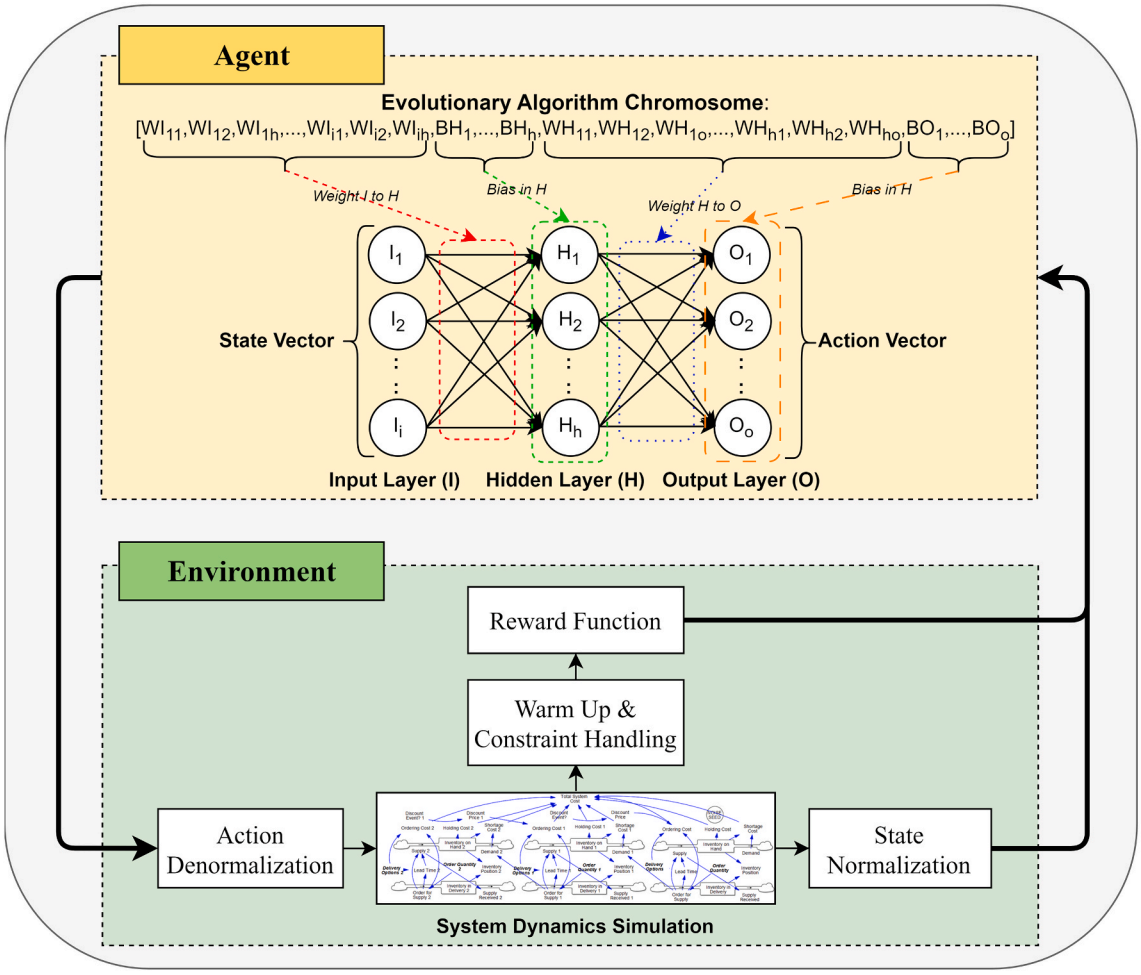


Fig. 2. Top level of proposed NERL framework.

to optimize the weights and biases of ANN, it is convenient to use EAs that work on continuous optimization space. Thus, three EAs will be experimented namely DE, MA, and ES. Each algorithm works as described in Fig. 3. It can be seen that each algorithm has different strategy in exploration and exploitation leading to the different performance of NERL.

In DE, for each individual $x_{i,t}$ with the first index representing the individual index (i = 1,2 …,N) and second index representing the generation index (t = 1,2, …,T), mutation process is first conducted by randomly selecting three other different individuals $x_{1,t}$, $x_{2,t}$, and $x_{3,t}$. DE mutation requires Equation (2) such that it generates a new candidate individual namely $v_{i,t+1}$ where F is a mutation scale constant. In order to increase the diversity, mix vectors $x_{i,t}$ and $v_{i,t+1}$ through uniform crossover generating new individual $x_{i,t+1}$ using Equation (3). This process happens for each variable index (j = 1,2, …,D with D = number of decision variables) and when a random number (rand = U [0,1]) is less than or equal a prespecified crossover probability (P_c). The fitness of $x_{i,t}$ and $x_{i,t+1}$ are evaluated and the selection process is conducted by applying greedy rule i.e. selecting the one with the better fitness value to the next generation.

Table 2
NERL pseudocode for MEIO.

Algorithm 1 Pseudocode of NERL for Multi-Echelon Inventory Optimization

Input: *EA Parameters: {Population size (N), Number of generations (T), Range of decision variables (R), Number of time steps (t_{max}), Number of episodes (E_{max})}, ANN Parameters: {Number of hidden layers (H) and hidden nodes (HN)}, and Problem Data*

Output: *W*: Set of optimum weights, B*: Set of optimum biases, Y*: Minimum Total Cost*

Steps:

Determine the states (S) and actions (A)
Configure ANN→ Input node = number of S, Output node = number of A, H, and HN.
Initialize EA population randomly within range (R) → Weights (W) and Biases (B) of ANN
for Generation $k = 1$ to T **do**
  for Individual $j = 1$ to N **do**
    Initialize expected reward (Y) of fitness j
    for episode $e = 1$ to E_{max} **do**
      Initialize total cost list (*TC_list*) with the length = E_{max}
      Implement *Common Random Number* by setting the seed = e
      for timestep $t = 1$ to t_{max} **do**
        if $t = 1$ **do**
          Initialize the inventory system randomly
        else
          Perform SD simulation at time t based on predicted actions
        end if
        Obtain the states *S(t)* and normalize them into 0–1 range
        Predict actions *A(t)* from ANN and denormalize them
      end for
      Cut the warm up period by excluding the costs in non-steady state from total cost
      Calculate the total cost and append it to *TC_list*
      if Inventory on hand for each facility > Maximum inventory capacity **do**
        Update total cost in *TC_List* by applying penalty-based constraint handling
      end if
      end for
    Measure Y by averaging *TC_list* as fitness value of j
  Perform genetic operators of EA to update individual chromosome
  Select the lowest $Y \rightarrow Y^*$ and store the optimal $W \rightarrow W^*$ and $B \rightarrow B^*$
  end for
end for
Return *W*, B*, Y**

In MA, it basically works by combining the global search capability of GA and local search. It starts by evaluating fitness function via simulation for each individual. Selection process based on roulette wheel is used to take two candidate individuals for matting and following genetic operators namely uniform crossover same as shown in Equation (3). MA mutation based on swapping process is then conducted which selects two positions on the individual chromosome randomly and exchange the values. Therefore, it requires P_c and mutation probability P_m as pre-specified parameters. After that, for each new individual will follow local search to further exploit neighborhood based on simple hill-climbing. For local search hyperparameters are set constantly namely probability of local search for each individual is set 0.5 and maximum local search is 10. The process repeats until new individuals as many as N. All processes are iterated until the maximum generation is reached.

The last algorithm is ES, specifically (μ + λ)-ES where μ represents population size and λ represents offspring size while + represents elitism strategy for selection. The main difference of ES compared to other two algorithms is that ES requires not only individual vector $x_{i,t}$ but also associated strategy parameter $\sigma_{i,t}$. While x can be randomly initialized, σ initialization is pre-determined parameter as well as the λ size. ES is started by randomly selecting 2 individuals from population to proceed local intermediate crossover as shown in Equation (4) for x and Equation (5) for σ. Afterward, ES mutation is conducted by using Equation (6) where individual vector is updated based on $\sigma_{i,t}$ with standard normal noise. The new generated solutions or offsprings are then combined with the existing individuals and fitness values are evaluated for all. Finally, the elitism selection process is done where the best fitness solutions will replace the existing population for the next generation. The selected solutions then replace the existing population and all processes are repeated until the maximum generation is reached.

$$v_{i,t+1} = x_{1,t} + F * \left(x_{2,t} - x_{3,t}\right) \tag{2}$$

$$x_{i,t+1} = \begin{cases} v_{j,i,t+1}, rand \leq P_c \\ x_{j,i,t}, rand > P_c \end{cases} \tag{3}$$

$$x_{i,t+1} = rand * x_{1,t} + (1 - rand)x_{2,t} \tag{4}$$

$$\sigma_{i,t+1} = rand * \sigma_{1,t} + (1 - rand)\sigma_{2,t} \tag{5}$$

$$x_{i,t+1} = x_{i,t} + \sigma_{i,t} * N(0,1) \tag{6}$$

ANN configuration also plays important role for NERL performance. The ANN consists of input layer, hidden layer, and output layer. In input layer, the number of input nodes depends on the number of states considered and the number of output nodes depends on the number of actions given. For hidden layer, and hidden nodes, both are pre-determined parameters. The larger the hidden layer and hidden nodes, the higher chance of ANN to capture complex relationship (Uzair and Jamil, 2020) leading to more optimal action similar as what Deep Reinforcement Learning (DRL) does. However, there will be a chance to be overfitting and the computation will be more expensive (Panchal and Panchal, 2014). Thus, both are hyperparameters that can be tuned. For solving MEIO-DO-UD, the states and actions for each facility are defined as follows.

- State: {Inventory position [Integer], Discount event by supplier [Binary]}
- Action: {Order quantity [Integer], Delivery options or mode chosen [Integer]}

3.2. Simulation modeling for multi-echelon inventory

Since the ultimate goal of this study is to solve dynamic complex multi-echelon inventory problem with multiple practical features, representing environment through direct mathematical model is complex to build and solve especially when many uncertainty factors considered. Thus, SD model is used imitating the multi-echelon inventory from retailer to the last supplier to simulate it over time. Thus, both environment and agent will interact as time step runs. Clearly, simulation model for multi-echelon inventory will be case specific depending on supply chain network itself. In this study, two generic simulation models have been built as presented in Figs. 4 and 5 representing serial and divergence networks, respectively. In both figures, only three facilities or companies are considered although it can be extended or reduced. These models have gone through verification via unit and error checking under peer-review manner and validation via face validation and parameter sensitivity techniques where it returns logically plausible under different model parameters.

In SD model, there are graphical representation consensus. Rectangles represent the Stock that accumulate or deplete the unit overtime. In this case, only 'Inventory on Hand' and 'Inventory in Delivery' variables belong to stock. The number behind the variable name only differs the facility. The arrows entering or exiting these stocks indicate Flows, which are quantities always expressed as rates of change. In this model, 'Demand', 'Supply', 'Order for Supply', and 'Supply Received' are identified as flows. Other variables are termed as auxiliary variables, provide a clearer representation of the factors influencing the flows in the system as indicated by blue arrows. Thus, SD does not only provide the quantitative model, but also qualitative model that helps stakeholders to understand the system easier. The main difference between

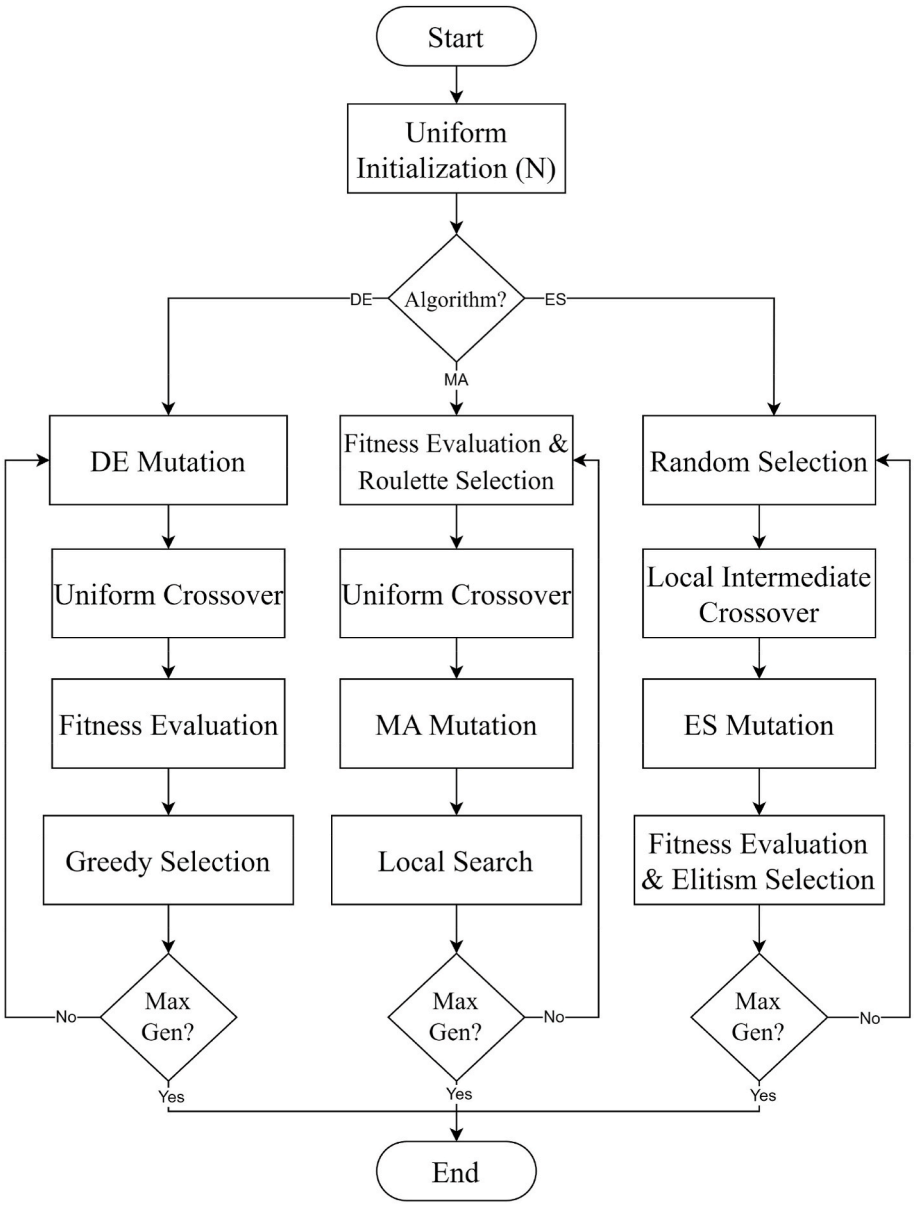


Fig. 3. Evolutionary algorithms used for MEIO

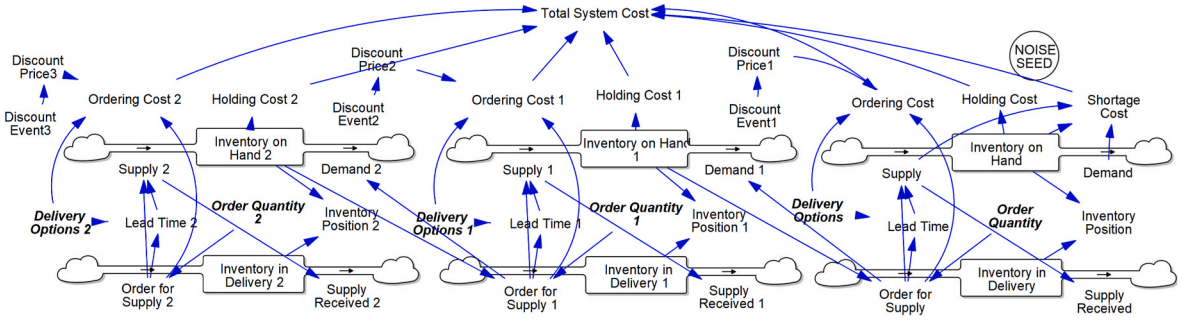


Fig. 4. SD model for serial multi-echelon inventory.

divergence and serial system is that, there is additional rule needed for divergence system represented by 'Inventory Checking' defining how the supplier allocates the items to the multiple facilities, especially when inventory on hand is limited.

In terms of relation between reinforcement learning and SD model, the states are represented by 'Inventory Position' and 'Discount Event' variables while actions are represented by 'Order Quantity' and 'Delivery Options' variables for each facility in each echelon. Since the expected reward function is important aspect in any reinforcement learning, thus, it is formulated mathematically as follows. The expected

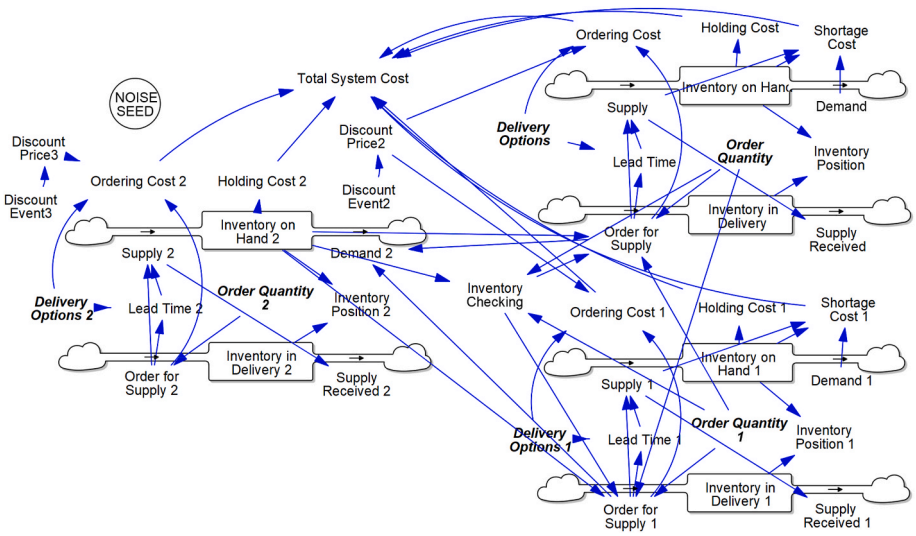


Fig. 5. SD model for Divergence multi-echelon inventory.

reward function representing total system cost can be formulated in Equation (7) measuring the total system cost. Since there are maximum inventory capacities for each facility as constraints, the constraint handling function is designed following penalty-based strategy. Therefore, the reward function in Equation (7) can be in two forms depending on whether constraints are violated or not. It basically measures total system cost that consists of the ordering cost as shown in Equation (8), holding cost as shown in Equation (9), shortage cost as shown in Equation (10), and M, where M represents large number. By doing so, the NERL algorithm will automatically look for the minimum solutions namely the ones without violated constraint.

It is worth to note that the ordering cost is affected by the delivery option chosen that drives the variable cost and transportation cost. Therefore, both costs are set as variable instead of fixed as many practical problems following this condition. In terms of shortage cost, since we want to make centralized decision among all echelons, the shortage cost is only considered in the last echelon which is directly associated with customers. Equation (11) ensures that the order quantity should be more than or equal to zero with only integer values allowed. Equations (12) and (13) define binary variables $f(s,i,t) = 1$ iff supplier s is assigned to supply facility i at time t and $DO(d,i,t) = 1$ iff delivery option d is chosen by facility i at time t, respectively. Further, for assessing the service quality, expected fill rate is also measured as shown in Equation (14). Finally, all equations are embedded in a SD model for solving multi-echelon inventory optimization. Note that the set of facility refers to all entities in supply chain while set of retailers only covers the entities that face the customer directly.

$$\mathbb{E}[\mathrm{TC}] = \begin{cases} min\,\mathbb{E}\left\{\sum_{t}^{T}\sum_{i}^{I}(OC(i,t)+\rho*IH(i,t))+\sum_{t}^{T}\sum_{r}^{R}(ST(r,t)*\gamma)\right\}, IH(i,t)\le IC(i) \\ min\,\mathbb{E}\left\{\sum_{t}^{T}\sum_{i}^{I}(OC(i,t)+\rho*IH(i,t))+\sum_{t}^{T}\sum_{r}^{R}(ST(r,t)*\gamma)+M\right\}, IH(i,t) > IC(i) \end{cases} \tag{7}$$

$$OC(i,t) = \sum_{t}^{T}\sum_{i}^{I}\sum_{s}^{S}\sum_{d}^{D}\{DO(d,i,t)*f(s,i,t)*\left(DC(s,t)*v(d)*\mathrm{Q}(i,t)+\left\lceil\frac{\mathrm{Q}(i,t)}{\mathrm{C}(d)}\right\rceil*A(d)\right)\} \tag{8}$$

$$IH(i,t) = IH(i,t_0)+\sum_{t_0}^{t}(D(i,t)\ -\mathscr{I}(i,t)) \tag{9}$$

$$ST(r,t) = \begin{cases} \mathscr{D}(i,t)-IH(r,t)+\mathscr{I}(i,t), \text{if } IH(r,t)+\mathscr{I}(i,t) < \mathscr{D}(i,t) \\ 0, Otherwise \end{cases} \tag{10}$$

$$\mathrm{Q}(i,t) \ge 0, \mathrm{Q}(i,t) \in \mathrm{Z}, \forall t \in T, i \in I \tag{11}$$

$$DO(d,i,t) \in \{0,1\}, \forall t \in \mathrm{T}, i \in I, d \in D \tag{12}$$

$$f(s,i,t) \in \{0,1\}, \forall t \in T, i \in I, \mathrm{s} \in S \tag{13}$$

$$\mathbb{E}[\mathrm{FR}] = \mathbb{E}\left\{\frac{\sum_{t}^{T}\sum_{r}^{R}\left(\frac{\mathscr{D}(r,t)-ST(r,t)}{\mathscr{D}(r,t)}\right)}{T*R}\right\} \tag{14}$$

4. Computational experiment

In this section, the proposed framework is implemented to solve the numerical examples, including how experiment is set up and the discussion of the results obtained. The cases given are realistic inventory problems consisting of multiple echelons, stochastic lead time, demand, and discount event, simultaneously considering delivery options, making the analytical approaches impractical to be used.

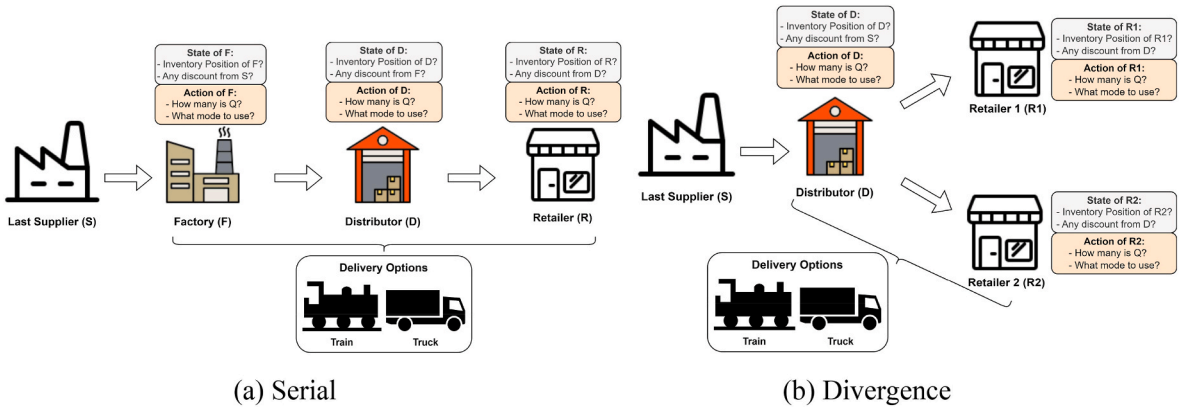


Fig. 6. Illustration of multi-echelon inventory networks with states and actions.

4.1. Case description

Since there is no existing literature on the MEIODOUD problem, no benchmark instances also exist. Thus, the problem instances were generated as follows. Since multi-echelon inventory complexity depends on the supply chain network, serial and divergence systems are given. In serial system, there are three echelons considered where each echelon only consists of one facility started from factory in the upstream, distributor, and retailer in the downstream. In divergence system, there are two echelons considered where the upstream echelon consists of one distributor and the downstream echelon consists of two retailers direct contact with customers. The two cases are illustrated in in Fig. 6.

Since the supply process is related to the supply chain network used, the supply allocation needs to be defined as follows. In serial network, when the order quantity from retailer is more than the supplier inventory on hand, the supplier only delivers the available quantities to retailers. In other words, it is not fully delivered and backorder is not allowed in suppliers. In divergence network, there is addition rule since there are multiple retailers. If the supplier has limited inventory on hand while the total demand from all retailers exceed, then the supplier will check first if any order quantity from retailers can be fulfilled fully from remaining inventory on hand of supplier with the largest order to be prioritized. Otherwise, the supplier's inventory will be delivered to retailers evenly. For the last supplier, it is assumed to always be able fulfilling all orders.

For simplicity the data used for both cases in each echelon are the same as described as follows. Holding cost is $2 per unit per day, stockout cost is $100 per shortage item, demand in all retailers follows independent and identically distributed (i.i.d) Poisson distribution with mean 15 (P (15)), and discount event has 10% chance per day. If there is a discount event at day t, then 15% discount is given to the variable cost (v). In terms of ordering cost, it is quantity (Q)-dependent and mode-dependent. Due to delivery options, it raises to the multiple transportation cost (A) and v, multiple lead time with different mean following Poisson distribution, and multiple maximum capacity (C) as presented in Table 3. In this case, 2 different delivery modes are considered. First is trucking option where it takes longer lead time due to road construction and traffic congestion delay but cheaper. Second, railing option where it takes faster lead time but more expensive. This is very practical situation where that trade-off always arises especially in long haul distribution.

Table 3
Delivery options data and cost components.

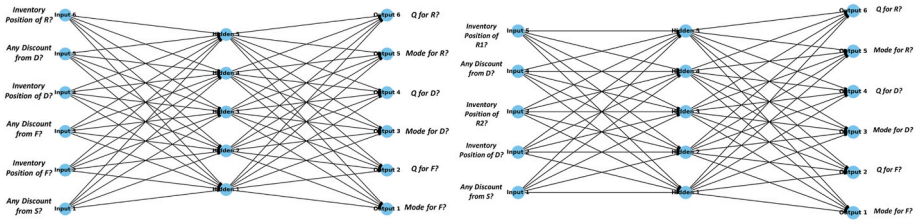
Truck ($C = 20$)		Rail ($C = 30$)	
Lead Time = P (5)		Lead Time = P (3)	
Q	Total Cost ($v_1 = 2$, $A_1 = 20$)	Q	Total Cost ($v_2 = 1.5$, $A_2 = 30$)
$0 < Q \leq 20$	$Qv_1 + A_1$	$0 < Q \leq 30$	$Qv_2 + A_2$
$20 < Q \leq 40$	$Qv_1 + 2A_1$	$30 < Q \leq 60$	$Qv_2 + 2A_2$
$40 < Q \leq 60$	$Qv_1 + 3A_1$	–	–

4.2. Experimental and parameter settings

Since ANN topology is case-specific, two different ANNs are created for further experiment as presented in Fig. 7, one for serial and another for divergence system. For serial case, there are 6 states and 6 actions needed. Thus, there are 6 input nodes and 6 output nodes. However, for divergence case, there are 5 states and 6 actions needed. This happens because the state of discount event given by distributor applies for two retailers. For all cases, only single hidden layer is considered with 5 hidden nodes in ANN. Therefore, the number of parameters is 71 and 66 for serial and divergence, respectively. The search space range for all weights and biases are set to be $[-2, 2]$. All activation functions used are logistics function which outputs the value between 0 and 1. Thus, denormalization is needed in order to obtain feasible actions. For order quantity action, the range is between 0 and 60 where if 0 is applied, then there is no order given. For the choosing delivery option as another action, the simple rule is applied namely if the output is less than 0.5, then choose the first mode. Otherwise, choose the second mode.

In terms of algorithms, since the performance of NERL also depend on the metaheuristic algorithm used, three algorithms belong to EA are experimented namely DE, MA, and ES. As comparison, apart from using NERL, those three EA algorithms are also applied for optimizing classical continuous review inventory method (s,Q) as baseline policy by directly optimizing reorder point (s) and order quantity (Q) for each echelon. Therefore, there will be six different algorithms used in total, namely three based on NERL policy abbreviated as DE-NERL, ES-NERL, and MA-NERL while the other three based on (s,Q) policy abbreviated as DE-(s, Q), ES-(s,Q), and MA-(s,Q).

Whether it is DE, MA, and ES, each has unique pre-determined hyperparameters. Therefore, parameter setting is necessary and preliminary experiment is conducted where each combination of parameters is experimented to optimize the baseline policy only. The parameter settings and the best set of parameters obtained for each metaheuristic algorithm are shown in Table 4. The best parameter is obtained through full factorial where each algorithm has 2 factors and 2 levels. The selected levels are based on suggestion from literature namely (Storn, 1996) for DE (Acampora et al., 2023), for MA, while ES is determined based on some fractions for simplicity.



(a) Serial (b) Divergence

Fig. 7. ANN topology for each problem.

Table 4
Parameter settings for all algorithms.

General Parameters	DE	ES	MA
N = 10	F = [0.5, 1]	Initial σ = [5%*R, 10%*R]	P_c = [0.7, 0.9]
T = 100	P_c = [0.8, 1]	λ = [0.5*μ, 0.75*μ]	P_m = [0.1, 0.15]
The Best Parameters	F = 0.5; P_c = 1	Initial σ = 5%; λ = 0.75*μ	P_c = 0.9; P_m = 0.15

* Population size (N); Number of generations (T); Mutation Scale Factor (F); Crossover Probability (P_c); Mutation Probability (P_m); Strategy Parameter (σ); Range of decision variables (R); Number of offsprings (λ). Number of parents (μ).

4.3. Computational results and discussion

The experiment was conducted in a computer with specification as follows. Processor: Intel® Core™ i7-6700 CPU @ 3.40 GHz (8 CPUs), Memory: 16 GB RAM, and Operating System: 64-bit Windows 10 Pro. SD models were built in Vensim software and NERL algorithms were built in Python. All algorithms are first trained using training data where each simulation evaluation takes 5 episodes. The trained policy is then tested under testing data. The performance comparison on training data is recapitulated in terms of expected reward (minimizing cost) measuring effectivity and training time measuring efficiency as presented in Table 5. The learning history is shown in Fig. 8 to see the behavior for each algorithm.

For training results, it can be assessed that in terms of effectivity, by comparing different algorithm in NERL, MA-NERL has the best performance in serial case but DE-NERL performs the best in divergence case. This implies that the metaheuristic algorithm is case-specific. In classical methods, MA-(s,Q) performs the best in all cases. Compared to the classical methods, no matter what optimizer is used, NERL always performs significantly better. In serial case, the best NERL algorithm performs 37.28% better than the best (s,Q) policy. In divergence case, the improvement becomes even more significant. The best NERL algorithm performs 57.93% better than the best (s,Q) policy. This implies that not in the more complex case, the need for applying NERL becomes more important.

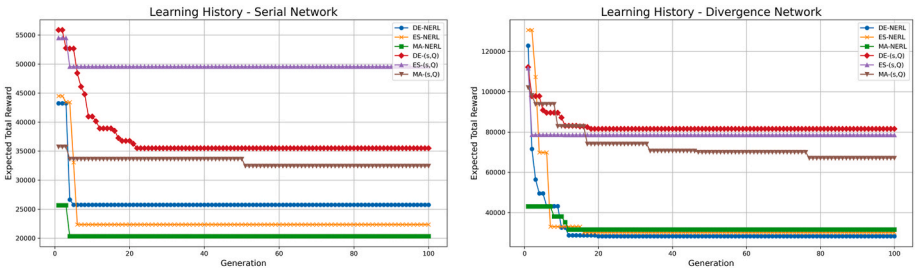
In terms of efficiency, NERL policy suffers from computational complexity compared to the (s,Q) policy. This is clearly because NERL performs dynamic optimization that provides different optimal decisions in each period while (s,Q) performs static optimization which only provides fixed optimal decisions for all periods. For metaheuristic algorithm efficiency, ES performs the best compared to MA and DE because ES has simpler genetic operators while MA becomes the most inefficient since it conducts not only global search but also local search leading to have more simulation evaluation. By seeing the learning history, MA always starts at the better performance in the first iteration due to local search operation. However, it seems that MA can easily get stuck to the local optimum. This implies that MA can be further improved by e.g. change the local search operation with the other algorithms e.g. Simulated Annealing (SA). DE has more stable convergence while but it is then stacked easily for DE-NERL. This implies that DE-NERL can be further improved by optimizing its parameter setting. ES seems to suffer from premature convergence with only small improvement in small iterations. Improving parameter setting may also improve ES performance.

Assessing the performance on training data only may lead to the overfitting chance. Thus, the performance assessment is also needed on

Table 5
Performance comparison on training data.

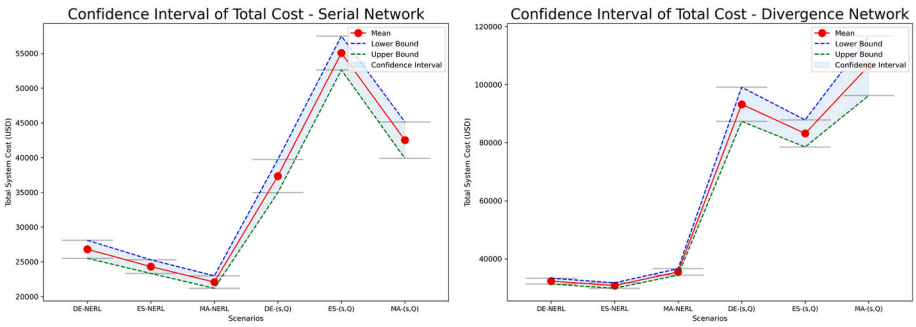
SC Network	Criteria	DE-NERL	ES- NERL	MA- NERL	DE-(s,Q)	ES-(s,Q)	MA-(s,Q)
Serial	Expected Reward	25,767	22,384	**20,341**	35,502	49,573	**32,431**
	Training Time (s)	10,650	**5942**	19,447	8411	**4164**	15,024
Divergence	Expected Reward	**28,207**	30,045	31,537	81,545	78,564	**67,050**
	Training Time (s)	12,005	**9231**	23,607	10,444	**7668**	17,924

*Bold values represent the best performance for each NERL and (s,Q) policies.



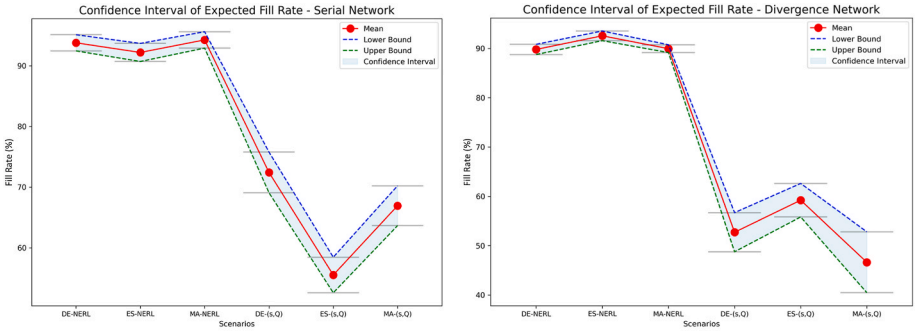
(a) Serial Network (b) Divergence Network

Fig. 8. Learning history in training Dataset.



(a) Serial Network (b) Divergence Network

Fig. 9. Confidence interval of expected total cost in testing Dataset.

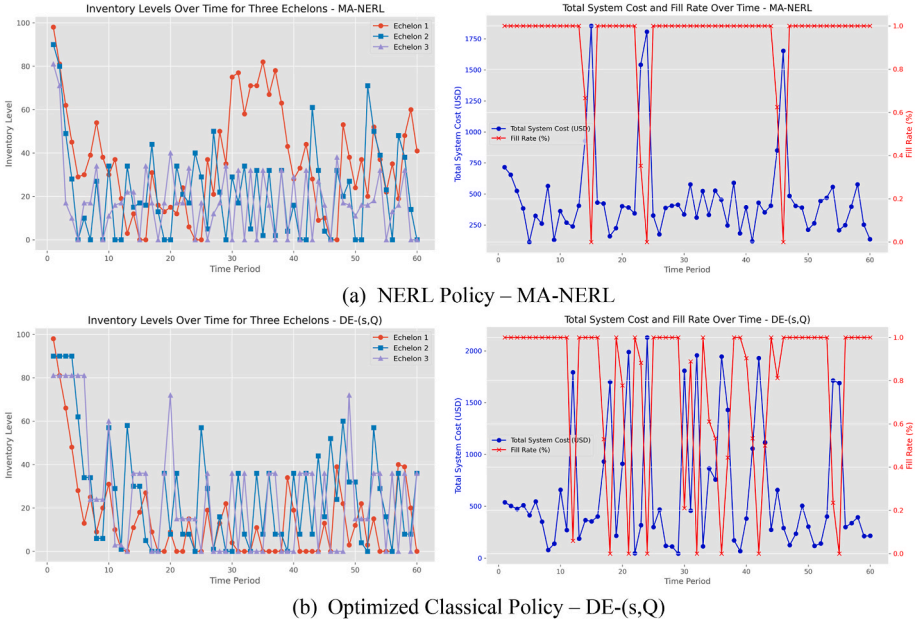


(a) Serial Network (b) Divergence Network

Fig. 10. Confidence interval of expected fill rate in testing Dataset.

the testing or unseen dataset. Since this study considers uncertainty, the expected total costs for each algorithm are compared and presented in terms of 95% t-distribution confidence intervals as shown in Fig. 9. It can be seen that in terms of the total cost, MA-NERL performs significantly better than others in serial case. This is accordance with the result obtained in training dataset implying MA-NERL can avoid overfitting. However, ES-NERL performs the best in divergence case while DE-NERL performs to be the second best even though the difference is overlapping or not statistically significant. This implies that DE-NERL suffers from overfitting since it has the best performance in training dataset. By comparing the confidence intervals of NERL policy with confidence intervals of (s,Q) policy, it also confirms that no matter what algorithm is used, NERL policy performs significantly better than (s,Q) policy statistically.

Further analysis is also conducted to see the impact on optimal decision to the expected fill rate as presented in Fig. 10. In serial case, it can



(a) NERL Policy – MA-NERL

(b) Optimized Classical Policy – DE-(s,Q)

Fig. 11. Inventory levels and performance measures of serial network.

be seen that the expected fill rates are around 91%–96% obtained by NERL policy and they perform significantly better than (s,Q) policy which have expected fill rates around 55%–75%. In divergence case, the expected fill rates obtained by NERL policy are around 89%–94% and they perform significantly better than (s,Q) policy either which have expected fill rates around 41%–63%. Thus, it has big gap of fill rate which can give huge impact to customer satisfaction. It also shows that the pattern obtained by expected total cost is similar with expected fill rate pattern. This is caused by relatively high stockout cost considered in this study. It is worth noting that since this problem is relatively new, no benchmark data is available limiting this result to specific examples only and may vary with different datasets e.g. when demand follows the time-dependent pattern. Nevertheless, it demonstrates the effectiveness of the algorithm under common and practical conditions.

4.4. Managerial implications

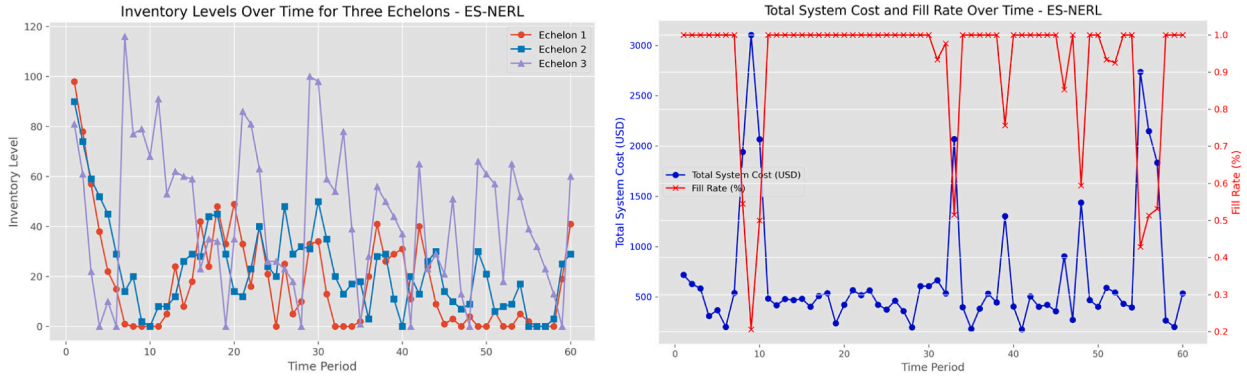
For generating managerial insights, we study the inventory level behavior for each echelon, including the dynamicity of total system cost and fill rate obtained by the best NERL policy and the best optimized classical inventory policy on testing data. For serial case, the results are shown in Fig. 11 where MA-NERL and DE-(s,Q) are selected as the best performance for each policy (NERL and continuous review inventory). Note that the first 10 days are not considered since it belongs to warm-up period. It can be seen that under stochastic demand, lead time, and discount event, MA-NERL can balance the inventory level for each echelon where the Echelon 1 representing retailer has relatively unstable inventory level due to high uncertainty of customer demand. However, this unstable inventory level can provide only small number of stockouts. Compared to the DE-(s,Q), it cannot balance the inventory level for all echelons where the Echelon 2 and 3 have relatively higher inventory level but retailer suffers from many stockouts.

For divergence case, the results are shown in Fig. 12 where ES-NERL and ES-(s,Q) are selected as the best performance for each policy. Clearly, divergence case is more complex than serial since two retailers have direct contact with customers. Therefore, it is plausible if it will have higher total cost and higher chance of stockout. It can be seen that ES-NERL has smaller chance to have stockout than ES-(s,Q). For both results, the Echelon 3 representing the distributor has significantly higher inventory level to fulfill both retailers. However, ES-NERL provides more stable inventory level for Echelon 3 without many sudden spikes like in ES-(s,Q).

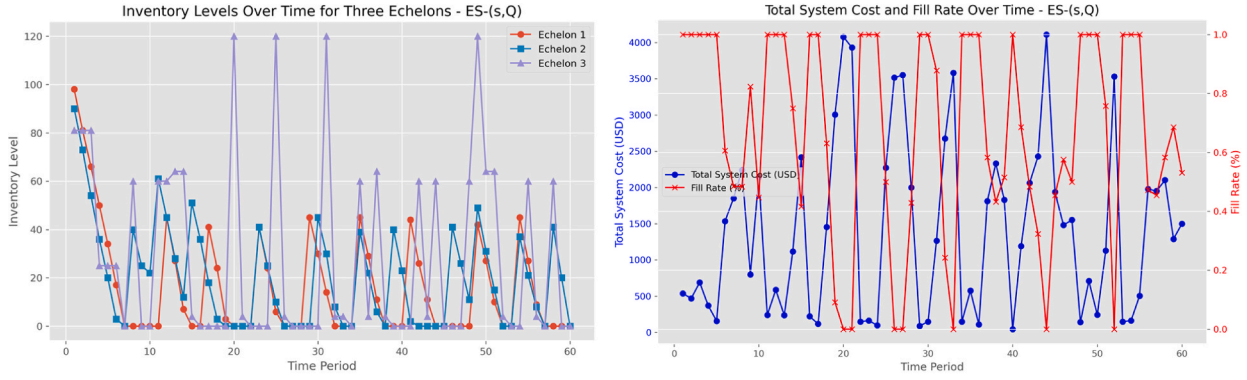
According to the serial and divergence cases discussed, it can be confirmed that NERL has better performance in giving more stable or balanced inventory level and reducing bull-whip effect among supply chain partners, in addition to provide lower total system cost, than classical continuous inventory review policy. In terms of applicability to the real world, the managers require to adapt to the advent of advanced information technology such as Radio-frequency identification (RFID) and Internet of Things (IoT), since it allows to provide real-time visibility of inventory states among supply chain partners, thus, enabling for more accurate and real-time inventory planning towards centralized and dynamic optimization.

5. Conclusion

This study introduces the new variant problem of inventory management so-called Multi-Echelon Inventory Optimization with Delivery Options and Uncertain Discount (MEIO-DO-UD). The Neuroevolution Reinforcement Learning (NERL) framework is also proposed for solving MEIO-DO-UD with the presence of stochastic demand and lead time with the objective for minimizing total system cost. NERL utilizes integration of ANN and EA as agent and System Dynamics simulation as environment. The systematic experiment is conducted by applying the proposed



(a) NERL Policy – ES-NERL



(b) Optimized Classical Policy – ES-(s,Q)

Fig. 12. Inventory levels and performance measures of Divergence network.

framework with three different metaheuristic algorithms namely DE, ES, and MA, to the two different supply chain networks as cases namely serial or divergence. The result shows that no matter what metaheuristic algorithm is used, the proposed framework always performs significantly better than the optimized classical continuous review inventory model for all cases. The proposed framework also has better performance not only in training dataset, but also testing dataset implying the capability for avoiding overfitting. The more complex problem i.e. divergence network, the more significant improvement can be made i.e. up to 58% cost reduction. Managerial implications are also highlighted in which the proposed framework provides the more stable inventory level among all supply chain partners and bull-whip effect can be damped. The reduction of total cost can also be followed by the improvement of fill rate. Further investigation is required to generalize the superiority of the NERL algorithm under different cases e.g. demand patterns. The introduced data can also be used for benchmarking purpose for the MEIO-DO-UD problems to reveal the best algorithm.

CRediT authorship contribution statement

Zakka Ugih Rizqi: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Data curation, Conceptualization. **Shuo-Yan Chou:** Supervision, Resources, Project administration, Funding acquisition, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data used are included within the article. The detailed findings will be made available on request. The simulation programs used are made available which can be accessed here: https://www.zakkaugihrizqi.com/projects/nerl.

Acknowledgments

This work was supported in part by the Intelligent Manufacturing Innovation Center from the Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education in Taiwan.

References

AbuZekry, A., 2019. Comparative study of NeuroEvolution algorithms in reinforcement learning for self-driving cars. European Journal of Engineering Science and Technology. https://doi.org/10.33422/ejest.2019.09.38.

Acampora, G., Chiatto, A., Vitiello, A., 2023. Training circuit-based quantum classifiers through memetic algorithms. Pattern Recogn. Lett. https://doi.org/10.1016/j.patrec.2023.04.008.

Almahamid, F., Grolinger, K., 2021. Reinforcement learning algorithms: an overview and classification. In: Canadian Conference on Electrical and Computer Engineering. https://doi.org/10.1109/CCECE53047.2021.9569056.

Anyibuofu, K.A., 2014. Inventory Management Practices in Manufacturing Firms. Industrial Engineering Letters.

Baioletti, M., Di Bari, G., Milani, A., Poggioni, V., 2020. Differential evolution for neural networks optimization. Mathematics. https://doi.org/10.3390/math8010069.

Barmi, Z.A., Ebrahimi, A.H., Feldt, R., 2011. Evolution strategies as a scalable alternative to reinforcement learning tim. In: 2011 IEEE Fourth International Conference on Software Testing, Verification and Validation Workshops.

Baruah, P., Chinnam, R.B., Korostelev, A., Dalkiran, E., 2016. Optimal soft-order revisions under demand and supply uncertainty and upstream information. Int. J. Prod. Econ. https://doi.org/10.1016/j.ijpe.2016.08.009.

Boute, R.N., Gijsbrechts, J., van Jaarsveld, W., Vanvuchelen, N., 2022. Deep reinforcement learning for inventory control: a roadmap. Eur. J. Oper. Res. https://doi.org/10.1016/j.ejor.2021.07.016.

Chandraju, S., Raviprasad, B., Chidan Kumar, C.S., 2012. Implementation of system application product (SAP) materials management (MM-Module) for material requirement planning (MRP) in sugar industry. Int. J. Sci. Res. Publ. 2 (9), 1–5.

Dittrich, M.A., Fohlmeister, S., 2021. A deep q-learning-based optimization of the inventory control in a linear process chain. J. Inst. Eng. Prod. https://doi.org/10.1007/s11740-020-01000-8.

Firoozi, Z., Tang, S.H., Ariafar, S., Ariffin, M.K.A.M., 2013. An optimization approach for A joint location inventory model considering quantity discount policy. Arabian J. Sci. Eng. https://doi.org/10.1007/s13369-012-0360-9.

Friedrich, L., Maziero, J., 2023. Evolution strategies: application in hybrid quantum-classical neural networks. Quant. Inf. Process. https://doi.org/10.1007/s11128-023-03876-8.

Gallego-garcía, D., Gallego-garcía, S., García-garcía, M., 2021a. An optimized system to reduce procurement risks and stock-outs: a simulation case study for a component manufacturer. Appl. Sci. https://doi.org/10.3390/app112110374.

García-Ródenas, R., Linares, L.J., López-Gómez, J.A., 2021b. Memetic algorithms for training feedforward neural networks: an approach based on gravitational search algorithm. Neural Comput. Appl. https://doi.org/10.1007/s00521-020-05131-y.

Geevers, K., van Hezewijk, L., Mes, M.R.K., 2022. Multi-echelon inventory optimization using deep reinforcement learning. SSRN Electron. J. https://doi.org/10.2139/ssrn.4227665.

Gijsbrechts, J., Boute, R.N., Van Mieghem, J.A., Zhang, D.J., 2022. Can deep reinforcement learning improve inventory management? Performance on lost sales, dual-sourcing, and multi-echelon problems. Manuf. Serv. Oper. Manag. https://doi.org/10.1287/msom.2021.1064.

Guo, S., Choi, T.M., Shen, B., Jung, S., 2019. Inventory management in mass customization operations: a review. IEEE Trans. Eng. Manag. https://doi.org/10.1109/TEM.2018.2839616.

Harris, F.W., 1990. How many parts to make at once. Oper. Res. https://doi.org/10.1287/opre.38.6.947.

Heidrich-Meisner, V., Igel, C., 2009. Neuroevolution strategies for episodic reinforcement learning. J. Algorithm. https://doi.org/10.1016/j.jalgor.2009.04.002.

Kara, A., Dogan, I., 2018. Reinforcement learning approaches for specifying ordering policies of perishable inventory systems. Expert Syst. Appl. https://doi.org/10.1016/j.eswa.2017.08.046.

Kaynov, I., van Knippenberg, M., Menkovski, V., van Breemen, A., van Jaarsveld, W., 2024. Deep reinforcement learning for one-warehouse multi-retailer inventory management. Int. J. Prod. Econ. https://doi.org/10.1016/j.ijpe.2023.109088.

Kosasih, E.E., Brintrup, A., 2022. Reinforcement learning provides a flexible approach for realistic supply chain safety stock optimisation. IFAC-PapersOnLine. https://doi.org/10.1016/j.ifacol.2022.09.609.

Kristiyani, I.M., Daryanto, Y., 2019. An inventory model considering all unit discount and carbon emissions. International Journal of Industrial Engineering and Engineering Management. https://doi.org/10.24002/ijieem.v1i2.3410.

Liu, X., Hu, M., Peng, Y., Yang, Y., 2022. Multi-agent deep reinforcement learning for multi-echelon inventory management. SSRN Electron. J. https://doi.org/10.2139/ssrn.4262186.

Mahapatra, A.S., Soni, H.N., Mahapatra, M.S., Sarkar, B., Majumder, S., 2021. A continuous review production-inventory system with a variable preparation time in a fuzzy random environment. Mathematics. https://doi.org/10.3390/math9070747.

Oroojlooyjadid, A., Nazari, M.R., Snyder, L.V., Takáč, M., 2022. A deep Q-network for the beer game: deep reinforcement learning for inventory optimization. Manuf. Serv. Oper. Manag. https://doi.org/10.1287/MSOM.2020.0939.

Panchal, F.S., Panchal, M., 2014. Review on methods of selecting number of hidden nodes in artificial neural network. Int. J. Comput. Sci. Mobile Comput. 3 (11), 455–464.

Patriarca, R., Di Gravio, G., Costantino, F., Tronci, M., 2020. EOQ inventory model for perishable products under uncertainty. J. Inst. Eng. Prod. https://doi.org/10.1007/s11740-020-00986-5.

Peng, Z., Zhang, Y., Feng, Y., Zhang, T., Wu, Z., Su, H., 2019. Deep reinforcement learning approach for capacitated supply chain optimization under demand uncertainty. In: Proceedings - 2019 Chinese Automation Congress (CAC). https://doi.org/10.1109/CAC48633.2019.8997498.

Prak, D., Teunter, R., 2019. A general method for addressing forecasting uncertainty in inventory models. Int. J. Forecast. https://doi.org/10.1016/j.ijforecast.2017.11.004.

Prestwich, S.D., Tarim, S.A., Rossi, R., Hnich, B., 2012. A neuroevolutionary approach to stochastic inventory control in multi-echelon systems. Int. J. Prod. Res. https://doi.org/10.1080/00207543.2011.574503.

Radaideh, M.I., Du, K., Seurin, P., Seyler, D., Gu, X., Wang, H., Shirvan, K., 2023. NEORL: NeuroEvolution optimization with reinforcement learning—applications to carbon-free energy systems. Nucl. Eng. Des. https://doi.org/10.1016/j.nucengdes.2023.112423.

Rekabi, S., Goodarzian, F., Garjan, H.S., Zare, F., Muñuzuri, J., Ali, I., 2023. A data-driven mathematical model to design a responsive-sustainable pharmaceutical supply chain network: a Benders decomposition approach. Ann. Oper. Res. https://doi.org/10.1007/s10479-023-05734-3.

Rizqi, Z.U., 2023. Capacitated continuous review inventory with partial backorder under time-dependent demand and fuzzy supply: Bi-objective optimization via simulation model. SSRN Electron. J. https://doi.org/10.2139/ssrn.4663662.

Rizqi, Z.U., Khairunisa, A., 2020. Integration of deterministic and probabilistic inventory methods to optimize the balance between overstock and stockout. In: IOP Conference Series: Materials Science and Engineering. https://doi.org/10.1088/1757-899X/722/1/012060.

Rizqi, Z.U., Khairunisa, A., Maulani, A., 2021. Financial assessment on designing inventory policy by considering demand, lead time, and defective product uncertainties: A monte carlo simulation. Indonesian Scholars Scientific Summit Taiwan Proceeding 3, 36–42. https://doi.org/10.52162/3.2021110.

Seyedan, M., Mafakheri, F., Wang, C., 2023. Order-up-to-level inventory optimization model using time-series demand forecasting with ensemble deep learning. Supply Chain Analytics. https://doi.org/10.1016/j.sca.2023.100024.

Shofa, M.J., Widyarto, W.O., 2017. Effective production control in an automotive industry: MRP vs. demand-driven MRP. In: AIP Conference Proceedings. https://doi.org/10.1063/1.4985449.

Singh, D., Verma, A., 2018. Inventory management in supply chain. In: Materials Today: Proceedings. https://doi.org/10.1016/j.matpr.2017.11.641.

Slowik, A., Kwasnicka, H., 2020. Evolutionary algorithms and their applications to engineering problems. Neural Comput. Appl. https://doi.org/10.1007/s00521-020-04832-8.

Storn, R., 1996. On the usage of differential evolution for function optimization. In: Biennial Conference of the North American Fuzzy Information Processing Society - NAFIPS. https://doi.org/10.1109/nafips.1996.534789.

Tunc, H., Kilic, O.A., Tarim, S.A., Eksioglu, B., 2011. The cost of using stationary inventory policies when demand is non-stationary. Omega. https://doi.org/10.1016/j.omega.2010.09.005.

Uzair, M., Jamil, N., 2020. Effects of hidden layers on the efficiency of neural networks. In: Proceedings - 2020 23rd IEEE International Multi-Topic Conference, INMIC 2020. https://doi.org/10.1109/INMIC50486.2020.9318195.

Wang, X., Wang, S., Liang, X., Zhao, D., Huang, J., Xu, X., Dai, B., Miao, Q., 2022. Deep reinforcement learning: a survey. IEEE Transact. Neural Networks Learn. Syst. https://doi.org/10.1109/TNNLS.2022.3207346.

Wu, G., de Carvalho Servia, M.Á., Mowbray, M., 2023. Distributional reinforcement learning for inventory management in multi-echelon supply chains. Digital Chemical Engineering. https://doi.org/10.1016/j.dche.2022.100073.

Zarandi, M.H.F., Moosavi, S.V., Zarinbal, M., 2013. A fuzzy reinforcement learning algorithm for inventory control in supply chains. Int. J. Adv. Manuf. Technol. https://doi.org/10.1007/s00170-012-4195-z.

Zhang, N., Gupta, A., Chen, Z., Ong, Y.S., 2022. Multitask neuroevolution for reinforcement learning with long and short episodes. IEEE Transactions on Cognitive and Developmental Systems. https://doi.org/10.1109/TCDS.2022.3221805.

Zhou, Y., Guo, K., Yu, C., Zhang, Z., 2024. Optimization of multi-echelon spare parts inventory systems using multi-agent deep reinforcement learning. Appl. Math. Model. https://doi.org/10.1016/j.apm.2023.10.039.

Zhu, C., Yang, B., Ma, H., Gao, C., Chen, J., 2022. Optimal strategy for a periodic review inventory system with discounted variable cost and finite ordering capacity. Oper. Res. https://doi.org/10.1051/ro/2022154.