Reinforcement Learning Approach for Multi-period Inventory with Stochastic Demand

Manoj Shakya[1](✉), Huey Yuen Ng[2], Darrell Joshua Ong[2], and Bu-Sung Lee[1]

[1] Nanyang Technological University, Singapore, Singapore
{manoj013,ebslee}@ntu.edu.sg

[2] Singapore Institute of Manufacturing Technology, Singapore, Singapore
nghy@simtech.a-star.edu.sg

Abstract. Finding an optimal solution to multi-period inventory ordering decision problems with uncertain demand is important for any manufacturing organization. Moreover, these problems are NP-hard as there are many factors to consider including customer demand and lead time which are stochastic in nature. This paper describes a reinforcement learning (RL) approach, Q-learning in particular, to decide on ordering policies. We formulated the finite horizon single-product multi-period problem into a reinforcement learning model in the form of Markov decision processes (MDP) and solve it to obtain the near-optimal solutions. Mixed integer linear programming (MILP) technique is still common in solving these problems; but they usually lack simplicity and may not optimized near to optimal. We formulated the same problem using the mixed integer linear programming model as the baseline algorithm so that we can compare it with RL approach. In comparison to MILP, the reinforcement learning agent performed better in making ordering decisions over the finite horizon. Obtaining better performance in multi-period problem would help the business in taking appropriate inventory decisions and reduce the total inventory costs.

Keywords: Reinforcement learning · Multi-period inventory management · Q-learning

1 Introduction

Optimal solution to inventory control and management problems is the crucial part of business solutions. On top of it, the stochastic inventory models have been the major focus of extensive research because the stochastic nature of variables make problem more challenging and complex. With the evolution of Industry 4.0, machine learning is playing important role in addressing such inventory control problems by optimizing the inventory costs [9].

Reinforcement learning is one of the techniques of machine learning (ML). There are generally three different threads in RL. The first one is *optimal control*; the second one is *trial and error*; and the third one is *temporal difference*. These three concepts form the basis of Reinforcement Learning (RL) [14]. Unlike supervised learning and unsupervised learning, the learner (agent) in reinforcement learning is not explicitly told to perform any particular action. In fact, in each time stamp, an agent closely observes the current state and takes an action so that returns it receives is maximum. Using the information of return, also called reward, an agent keeps on updating the knowledge of environment and selects the next possible action. It is a way to map situations to actions so that the environment maximizes a reward value.

In 2002, reinforcement learning was used in solving inventory optimization [3]. Later in 2008, RL techniques were used to solve the beer game problem [2]. Beer game problem is a popular simulation tool for the study of supply chain management that depicts a bullwhip effect. In that study, the Q-learning algorithm [14] and the genetic algorithm (GA) based algorithm were compared. In 2015, a deep-Q-network algorithm was developed making deep RL strong enough to solve many sequential decision making problems [8]. In 2017, deep RL was also introduced to solve beer game problem in supply chain management [9].

In recent years, RL has evolved to handle various supply chain management problems. It was used in addressing the coordination problem of global supply chain management [12]. The model called SMART (Semi Markov Average Reward Technique) was proposed. Considering a general supply chain, the coordination of inventory policies adopted by different agents, such as suppliers, retailers, manufacturers is a major issue. All these agents need to coordinate to minimize the total inventory costs while meeting the customer demand. In [3], RL approach was used to determine and manage the inventory decisions at all stages aiming at optimizing the performance of supply chain. [6] uses approximate SARSA (State Action Reward State Action) and REINFORCE algorithms to solve a supply chain optimization problem which is very much similar to the problem we've considered.

In [2], Q-learning algorithm was proposed to optimize inventory order decisions of four-stage supply chain. Similarly, a research work [5] has used Q-learning and State-action-reward-state-action (SARSA) algorithms and managed to minimize total cost of a retailer when dealing with the inventory management system of perishable products under the random demand and deterministic lead time. The research work by Oroojlooyjadid et al. [9] proposed an RL approach based on Deep Q-Network (DQN) and a transfer learning to calculate the optimal ordering policy. Limited research [4,9–11] have applied deep reinforcement learning models to inventory management problems (for example beer distribution game and newsvendor problems). A paper by Bharti S. et al. [1] applied Q-learning algorithm to solve single agent supply chain problem that is related to ordering decision problem. It is found that Q-learning approach is better than Order-Up-To (OUT) policy and 1-1 policy [1].

Although many studies deal with bullwhip effect and examine how RL and DRL techniques deal with beer game problem, a few studies are made to

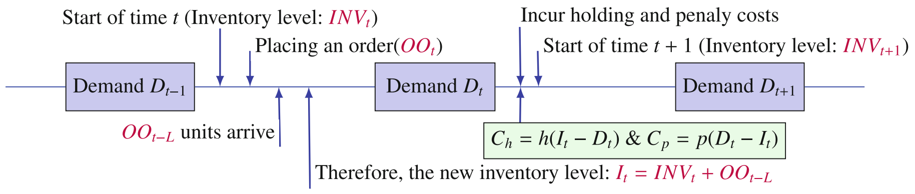

Start of time t (Inventory level: INV_t)
Placing an order(OO_t)
Incur holding and penaly costs
Start of time $t+1$ (Inventory level: INV_{t+1})
Demand D_{t-1}
Demand D_t
Demand D_{t+1}
OO_{t-L} units arrive
$C_h = h(I_t - D_t)$ & $C_p = p(D_t - I_t)$
Therefore, the new inventory level: $I_t = INV_t + OO_{t-L}$

Fig. 1. Timing of the sequential events

understand how RL handle multi-period inventory management problem [13]. This paper contributes on identifying the effectiveness of Q-learning algorithm for the multi-period inventory management problem taking in consideration the stochastic demand and deterministic lead time. In this paper, we have considered the MILP model as baseline as it is one of the popular operation research (OR) approaches and compared the results with that of reinforcement learning approach.

2 Problem Description and Modeling

2.1 Problem Description

We consider a single product and a periodic-review stochastic inventory control system with lost sales and positive lead times.

The event timing of the problem description is depicted in Fig. 1. An inventory manager makes sequential decisions in discrete time steps $t = 1, ..., T$. In the beginning of every time step t, (every month in our case), inventory manager observes the current inventory level INV_t, and open-order level that are unfulfilled orders in the pipeline. Let's denote this open-order as $OO_{t-L}, ..., OO_{t-1}$. Here, $L > 0$ is the lead time which is defined as the duration between placing an order and receiving it. Based on the inventory and order level, inventory manager decides on the amount to be ordered in the current time step t. Note that the order placed is received L time steps later. After placing the order and receiving the items ordered previously, the on-hand inventory at time step t will be $I_t = INV_t + OO_{t-L}$. Now, the inventory manager observes the demand $D_t \geq 0$. If the demand is more than on-hand inventory (i.e. $D_t > I_t$), the lost sales is recorded and penalty cost is incurred. If $I_t > D_t$, a holding cost of $C_h = h(I_t - D_t)$ incurs otherwise a penalty cost of $C_p = p(D_t - I_t)$ is incurred on the part of demand that could not be met due to insufficient on-hand inventory. Therefore, the total cost incurred at the end of time step t can be expressed as $C_t = C_h + C_p = h(I_t - D_t)^+ + p(D_t - I_t)^+$, where $(I_t - D_t)^+ = max(I_t - D_t, 0)$, $(D_t - I_t)^+ = max(D_t - I_t, 0)$, and h, p are pre-specified constants denoting per unit holding cost, and per unit penalty cost respectively.

The Fig. 1 illustrates the sequence of events that is explained above. The next step $t+1$ begins with the leftover inventory: $INV_{t+1} = (INV_t + OO_{t-L} - D_t)^+$ and the new pipeline of open-order will be: $OO_{t-L+1}, ..., OO_t$.

The objective of inventory control management is to find the policy that an agent should follow so that the total inventory cost of the system consisting of holding cost and penalty cost with lost sales is minimized.

2.2 MDP Formulation

To solve the aforementioned problem, we represent it with Markov decision processes (MDP) [14]. MDP can be expressed as (S, A, Pr, R_a) where S is the set of states, A is the set of actions, Pr is the transition probabilities, and R_a is the reward. Since the Q-learning is model free algorithm, we do not need to consider transitional probabilities.

Decision epochs: In this problem, the length of the timeline is 12 months. $t = \{1, 2, ..., 12\}$.

States: The system state variable provides the important information to the agent so that an agent can make optimal sequential decision in each step. Since the capacity of the store is M, and backlog order is not maintained, a set of states in this problem is the combination of inventory level, lost sales, open-order, and the order received at that time step (we call it shipment received - SR). It can be stated as $S = \{(INV_t, LS_t, OO_{t-L}, SR_t)\}$.

Action: Action set is a set of number that represents the order that can be placed at each time step t. In the beginning of every time step, an action is taken. Based on the assumptions a set of action can be expressed as: $A = 0, 1, 2, ..., M$. Theoretically, the demand can be of any size. If there were no capacity constrains, the set of actions would have been $A = 0, 1, 2, ..., M$. Since we have a limit to capacity of storing items in inventory, the set of action will be $A = \{0, 1, 2, ..., \alpha\}$ where $\alpha = (M - I_t - OO_t)$.

Reward: Since the main objective is to minimize the total inventory cost, the reward can be stated as $r_t(s, a) = h(I_t - D_t)^+ + p(D_t - I_t)^+$. Here, r_t denotes the net reward at time step t, h is unit holding cost, and p is the unit penalty cost.

2.3 Modeling with Q-learning

The state of the environment initially will be (INV, LS, OO, SR) as we initially have zero inventory level, no lost sales, and no any orders in pipeline. The agent (inventory manager) takes an action. i.e. the agent places an order to the supplier. This order OO_t will be appended in transit as an open order because of the lead time. The action/order an agent places follows the exploration and exploitation phenomenon. In exploration process, an agent randomly picks a number OO_t such that $OO_t \in\ A$. In exploitation process, an agent uses Q-table to get the action so that the reward for that action is maximum. But how does an agent decide which path to follow? Generally an ϵ value (exploration rate) is defined in the environment where $\epsilon \in [0, 1]$. A threshold value is randomly generated. If the threshold value is greater than the ϵ then the agent will exploit the environment and choose the action (i.e. places the order) that has the highest Q-value in the Q-table at time step t. If, on the other hand, the threshold is less than or equal to the ϵ, then the agent will explore the environment, and picks one value (randomly) from the action space.

As the agent performs an action of ordering products from supplier, there incurs costs like ordering cost, purchasing costs and so on. Besides these costs, there are other inventory costs that include holding cost, penalty cost, spoilage cost, and transportation cost etc. But we only consider holding cost and penalty cost in our optimization problem because these are directly related to over-stocking and under-stocking situations. After agent places an order, an environment receives the shipments from supplier. The total inventory level at this moment is $INT_t + OO_{t-L}$. Since we are not serving the backlog order, the new inventory level before serving the demand will be $I_t = (INV_t + OO_{t-L})^+$. We then serve the demand D_t. After the demand D_t is served, the reward is evaluated. By the end of the time step t, in our case a month, a reward can be calculated.

The essence of Bellman equation is to find the optimal policy that can produce the best action at any given state. The basic idea behind the Bellman equation is that the value of a current state is the sum of reward of being in that state and the reward you will be receiving after visiting subsequent states. [13]

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + ... = \sum_{k=0}^{T} \gamma^k r_{t+k+1} \tag{1}$$

where the discount faction $\gamma \in [0, 1]$. After we observe the reward that we had received taking the action from the previous state, we can update the Q-values for the state-action pair in the Q-table. We use the following Bellman equation to update the Q-value [13]:

$$Q^*(s,a) = Q(s,a) + \alpha\bigg(-R_t + \gamma \max_{a'} Q(s',a) - Q(s,a)\bigg) \tag{2}$$

where $Q^*(s,a)$ represents the action-value function that produces optimal policy, α is the learning rate and γ is the discount factor. The negative reward $(-R_t)$ is used because we want to minimize the cumulative reward value.

2.4 Modeling with MILP

We formulate the same aforementioned problem using MILP. Specifically, MILP is often used for solving optimization problem because it can offer flexible and powerful method to solve some complex problems like inventory management problem [7]. The method that is used here for comparison is a customized inventory planning algorithm that has applications in various industries. Its purpose is to optimize inventory of materials so as to achieve minimal inventory cost, minimal material wastage and maximize customer service level.

The inventory planning algorithm is based on stochastic programming, and the decision variables are the order quantity, and supplier choice over the specified planning horizon. In the model, we optimize the decision variables for multiple materials. There are S suppliers which supply the material m. The model solves an inventory planning problem with a finite planning horizon T which is composed of T planning periods, starting with period $t = 1$ and ending with period $t = T$.

For this experiment, the number of supplies considered is 1, and the number of materials passed to the model is 1. The demand over the planning horizon T is deterministic. The stochastic optimization model considers a finite planning horizon T which is composed of T planning periods, starting with period $t = 1$ and ending with period $t = T$.

3 Experimental Results

Once the training of RL agent was over, we evaluated the performance of the RL model. This section also explains the initial state of the retailer, i.e. the initial inventory level, unit of lost sales, open-order, and shipment received when starting the evaluation.

3.1 Training the Model

In order to train the RL model, we set the parameters as listed in the following Table 1.

Table 1. Parameter values used in training the Q-learning Agent

Parameters	Values
Maximum capacity of a store M	10
Holding cost per unit h	4
Penalty cost per unit p	8
Initial Inventory Level INV	0
Initial Lost Sales $lostsales$	0
Initial open-order $openorder$	0
Initial shipment received SR	0
Lead Time L	2

Table 2. Hyper-parameter values used in training the Q-learning Agent

Parameters	Values
Learning rate α	0.001
Discount factor γ	0.95
Initial exploration rate - ϵ	1.0
Exploration decay rate	0.001
Maximum episode	1e5
Time period T	12

We also set the hyper-parameters for training the RL agent. Table 2 shows the hyper-parameters used during the implementation of Q-learning. These hyper-parameters are selected because it provided the lowest average reward during training phase. After setting the parameters and hyper-parameters, we train the RL agent. The dataset is synthetic and generated using demand distribution. During the training phase, at each time step t, an action is predicted based on exploration value, demand is realised and finally the reward is calculated. This continues till $t = 12$ an then the environment is reset. This is a single rollout also called an episode. An agent was trained with 1e5 episodes.

	a_1	a_2	...	a_m
s_1	$Q(s_1,a_1)$	$Q(s_1,a_2)$	...	$Q(s_1,a_m)$
s_2	$Q(s_2,a_1)$	$Q(s_2,a_2)$	...	$Q(s_2,a_m)$
...	...	...	...	...
s_n	$Q(s_n,a_1)$	$Q(s_n,a\_2)$	...	$Q(s_n,a_m)$

(a) Q-table

	0	1	2	**3**	4	5
7318	-9648	-9711	-9686	-9371	-5456	-9700
7319	-9656	-9599	-9682	**-4993**	-9703	-9644
7320	-9559	-9580	-9540	-9529	-9578	-9578

(b) Q-values

Fig. 2. Q-table with Q-values

3.2 Agent's Policy

In this experiment, an optimal policy is learned when Q-table is updated subsequently. The value in the cell $Q(s,a)$ represents the value of the action a taken when the state is s. Given a state s, the action a that has highest $Q(s,a)$ is selected. As the iteration during training continues, $Q(s,a)$ for all combination of s,a are calculated with greedy search approach. With this approach, Q-values are converged to the near optimal policy. The Fig. 2a helps us to visualise the Q-function as a simple Q-table.

The state S_t of our environment is the combination of four variables. They are inventory level, backorder, open-order, and shipment received. With the values of these variables, a code (index in this case) is generated using a function. This code represents the row in a Q-table.

Solving a problem using reinforcement learning means finding a policy that provides the maximum reward during the training phase. Hence, the policy of an agent is the corresponding column value of the maximum Q-value in the row i.e. $a = maxQ(s,a)$ where a is the amount to be ordered when state s is reached. Let's take an example of state with inventory level: 2, lost sales: 0, open-order: 9, and shipment received: 4. The state of the environment is $S(2,0,9,4)$ which gives a code/index as 7319. Now, looking at the Q-table, the maximum Q-value in a row 7319 is located at $Q(7319,3)$. Hence, the policy suggests to place an order of 3 (#column). The Fig. 2b is a snapshot of the Q-table we obtained after training an agent. After placing an order of 3, and the realization of demand is 2, the new state of the environment will be $S(0,0,8,5)$. In the next step, the above procedure is repeated.

3.3 Initial Setting for Evaluation

The demand distribution is shown in Table 3. Let's set that the initial state of the inventory has Inventory $(INV_0) = 10$ units, Lost sales $(LS_0) = 0$, Open order for the first month $(OO_0) = 4$ units, and Open order for the second month $(SR_0) = 5$ units.

We examined the RL agent with 10 different test datasets, each consisting of randomly generated demands with finite planning horizon of $T = 12$ as shown in Table 4. The same test datasets were also being used in MILP model. As shown in Table 4, RL agent performed better than MILP agent in those test datasets. Let's consider the first test dataset out of 10 listed in Table 4 to see

Table 3. Demand distribution

Demand	Cumulative percentile
3	0.25
4	0.75
6	0.95
7	1.00

Table 4. Comparison of inventory cost generated by RL agent and MILP agent from 10 different test datasets.

Test dataset	List of demand	Total cost (RL Agent)	Total cost (MILP)
1	[6 7 6 6 4 3 6 4 4 4 3 3]	**92**	108
2	[6 7 6 6 3 6 4 4 4 4 4 4]	**96**	120
3	[6 6 7 6 3 3 3 7 4 4 3 4]	**88**	116
4	[6 7 6 7 4 4 4 3 3 6 4 6]	**124**	136
5	[6 7 6 6 4 4 6 3 4 3 3 4]	**84**	100
6	[6 6 6 6 3 3 4 3 4 4 6 6]	**96**	128
7	[6 7 6 6 4 4 6 3 3 4 3 4]	**88**	100
8	[6 6 6 6 4 3 3 4 6 3 3 4]	**96**	112
9	[6 7 6 6 4 4 6 3 3 3 3 4]	**84**	96
10	[6 7 6 6 3 6 4 4 4 4 4 6]	**88**	128

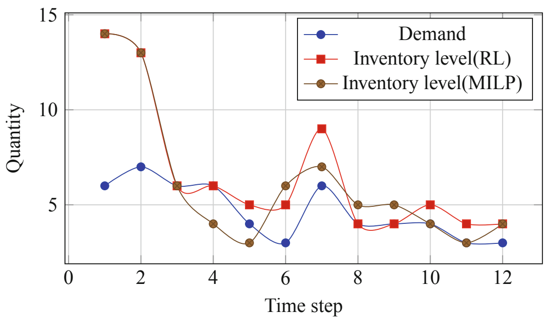
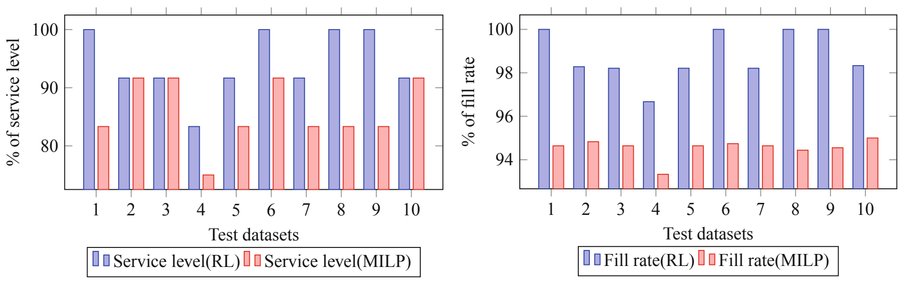


Fig. 3. Inventory level maintained by RL and MILP agents over demand

how inventory levels are maintained. Inventory level is a total stock in hand at a particular time step. Besides total cost, other indicators are also important. Figure 3 shows the inventory level of the first test dataset.

Some of the Key Performance Indicators (KPIs) in inventory management other than total inventory costs are *service level* and *fill rate*. Service level measures the percentage of not getting stock-out i.e. not losing the sales of all customers' demand arriving within a given planning horizon. For example, if the current service level is 80%, it means there are two time steps out of 10 where customers' demand could not be fulfilled. Let D_t be the demand over the time period t and I_t be the current inventory level. Then service level (η_1) is:

$$\eta_1 = \frac{1}{T} \sum_{t=1}^{T} S \qquad (3)$$



(a) Service Level (RL vs MILP) (b) Service Level (RL vs MILP)

Fig. 4. Service Level and Fill Rate

where $S = 1$ if stock is available to fulfill the customers' demand, and $S = 0$ otherwise. Unlike service level, fill rate is the percentage of customers' demand that is fulfilled without lost sales. Service level calculates the fraction of replenishment cycle during which customers' demand are fulfilled whereas fill rate calculates the fraction of demand satisfied from available stock. For instance, 98% fill rate means, it fulfilled 98 customers' demand out of 100. The formula to calculate the fill rate (η_2) is:

$$\eta_2 = \frac{\sum_{t=1}^{T} min(D_t, I_t)}{\sum_{t=1}^{T} D_t} \tag{4}$$

Figure 4a and Fig. 4b respectively show the service levels and fill rates maintained by RL and MILP agents in all 10 different test datasets listed in Table 4.

In most of the test datasets, RL agent successfully achieved 100% of service level where as MILP agent struggles to maintain 90% service level. This signifies that RL agent could do better in predicting the future demand and could take an account of supply and demand variance very well. Similarly, most of the time the fill rate obtained by MILP agent is lower than the one obtained by RL agent. The better fill rate obtained by RL agent signifies that the ordering decisions made by RL agent could meet customers' demand very well.

4 Conclusions and Future Work

This paper models the multi-period inventory management problem with stochastic demand into Markov decision processes and solves it using value-based approach of reinforcement learning so called Q-learning. The RL approach of solving the aforementioned problem has shown good result. The RL model in comparison with MILP model shows that it can handle the fluctuation of demand with high service level and good fill rate. In addition, the exploratory results presented in this paper proves that RL can be a good approach for solving multi-period inventory control that adds a great value to smart manufacturing processes. In future, research can be conducted to solve complex multi-period inventory control problems introducing more business and operational constrains.

Acknowledgements. The authors would like to thank the Singapore Institute of Manufacturing Technology (SIMTech), Singapore for its input and support.

References

1. Bharti, S., Kurian, D.S., Pillai, V.M.: Reinforcement learning for inventory management. In: Deepak, B.B.V.L., Parhi, D.R.K., Jena, P.C. (eds.) Innovative Product Design and Intelligent Manufacturing Systems. LNME, pp. 877–885. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-2696-1_85
2. Chaharsooghi, S.K., Heydari, J., Zegordi, S.H.: A reinforcement learning model for supply chain ordering management: an application to the beer game. Decis. Support Syst. **45**(4), 949–959 (2008)
3. Giannoccaro, I., Pontrandolfo, P.: Inventory management in supply chains: a reinforcement learning approach. Int. J. Prod. Econ. **78**(2), 153–161 (2002)
4. Gijsbrechts, J., Boute, R.N., Van Mieghem, J.A., Zhang, D.: Can deep reinforcement learning improve inventory management? Performance and implementation of dual sourcing-mode problems. SSRN Electron. J .1–26 (2019)
5. Kara, A., Dogan, I.: Reinforcement learning approaches for specifying ordering policies of perishable inventory systems. Expert Syst. Appl. **91**, 150–158 (2018)
6. Kemmer, L., Read, J.: Reinforcement learning for supply chain optimization. In: The 14th European Workshop on Reinforcement Learning, vol. 14 (2018)
7. Küçükyavuz, S.: Mixed-integer optimization approaches for deterministic and stochastic inventory management. In: Transforming Research into Action, pp. 90–105. INFORMS (2011)
8. Mnih, V., et al.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529–533 (2015)
9. Oroojlooyjadid, A., Snyder, L., Takáč, M.: A Deep Q-Network for the Beer Game: an Approach to Solve Inventory Optimization Problems. Deep Reinforcement Learning Symposium, NIPS 2017 (2017)
10. Oroojlooyjadid, A., Snyder, L.V., Takáč, M.: Applying deep learning to the newsvendor problem. IISE Trans. **52**(4), 444–463 (2020)
11. Peng, Z., Zhang, Y., Feng, Y., Zhang, T., Wu, Z., Su, H.: Deep reinforcement learning approach for capacitated supply chain optimization under demand uncertainty. In: Proceedings - Chinese Automation Congress, pp. 3512–3517 (2019)
12. Pontrandolfo, P., Gosavi, A., Okogbaa, O.G., Das, T.K.: Global supply chain management: a reinforcement learning approach. Int. J. Prod. Res. **40**(6), 1299–1317 (2002)
13. Sultana, N.N., Meisheri, H., Baniwal, V., Nath, S., Ravindran, B., Khadilkar, H.: Reinforcement Learning for Multi-Product Multi-Node Inventory Management in Supply Chains. CoRR (2020). http://arxiv.org/abs/2006.04037
14. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. The MIT Press, London, England (2018)