Decision Support Systems 45 (2008) 949–959

Contents lists available at ScienceDirect

Decision Support Systems

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

A reinforcement learning model for supply chain ordering management: An application to the beer game

S. Kamal Chaharsooghi *, Jafar Heydari, S. Hessameddin Zegordi

Industrial Engineering Department, School of Engineering, Tarbiat Modares University, Tehran, Iran

ARTICLE INFO

Article history:
Received 18 July 2006
Received in revised form 18 March 2008
Accepted 26 March 2008
Available online 8 April 2008

Keywords:
Supply chain
Ordering policy
Multi-agent systems
Beer game
Reinforcement learning

ABSTRACT

A major challenge in supply chain ordering management is the coordination of ordering policies adopted by each level of the chain, so as to minimize inventory costs. This paper describes a new approach to decide on ordering policies of supply chain members in an integrated manner. In the first step supply chain ordering management has been considered as a multi-agent system and formulated as a reinforcement learning (RL) model. In the final step a *Q*-learning algorithm is proposed to solve the RL model. Results show that the reinforcement learning ordering mechanism (RLOM) is better than two other known algorithms.

1. Introduction

Supply chain management (SCM) literature covers wide range of areas such as logistics, production, scheduling, facility location, procurement, inventory management, ordering management, and so on. Supply chain ordering management (SCOM), which is the main concern of this paper is an integrated approach to determine the ordering size of each actor of SC to the upstream actor aiming to minimize inventory costs of the whole supply chain. SCOM is ultimately focused on the demand of the chain aiming to reduce inventory holding costs, lower slacks, improve customer services, and increase the benefits throughout the entire supply chain.

In this paper, the supply chain is considered as a combination of various multi-agent systems collaborating with each other. Thus, SCOM can be viewed as a multi-agent system, consisting of ordering agents. Each ordering agent tries to make decisions on ordering size of the relevant echelon by considering the entire supply chain. Agents interact and cooperate with each other based on a common goal. For example, in a linear supply chain with four echelons (as considered in this paper), there are four ordering agents in SCOM system, each of which is responsible for ordering decisions in its particular echelon. The main objective of ordering agents is to minimize long-term system-wide total inventory cost of ordering from immediate supplier. This is a complex task because of the uncertainty embedded in the system parameters (e.g. customer demand and lead-times) and demand amplification effect [4], known as 'bullwhip effect'[13].

This paper has focused on the ordering agents of the supply chain and aims to make a proper learning mechanism for these agents. Under learning mechanism, agents learn how to react to the changing environment.

The type of considered supply chain is serial with four levels: retailer, distributor, manufacturer, and supplier respectively. A classical example of supply chain ordering management is the MIT beer game [24], which has attracted much attention from supply chain management researchers. In the MIT beer game, actor of each level attempts to minimize the whole supply chain inventory costs. The decision of ordering size depends on various factors such as supply chain inventory level, environment parameters, downstream ordering size, and so on. Since, companies face global markets and highly turbulent environments, complexity of

* Corresponding author. Tel.:+98 21 44209944.
E-mail addresses: SKCH@modares.ac.ir (S.K. Chaharsooghi), Heidarij@modares.ac.ir (J. Heydari), Zegordi@modares.ac.ir (S.H. Zegordi).

0167-9236/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.dss.2008.03.007

production and business processes is increased. In such complex conditions a fixed ordering rule cannot achieve the system's goal and therefore supply chain actors must make their decisions based on the system's state. Previous related works proposed fixed ordering rules with no attention to the uncertainties of the environmental factors and their impacts on ordering policy of the chain. The current research addresses this problem by considering environment state in producing ordering policy in each time step. In our proposed model, environmental uncertainties include customer demand and lead-times as two common uncertainties in real world supply chains.

The paper is organized in the following way. Section 2 provides a brief literature review. Section 3 generally describes the reinforcement learning problem. Section 4 describes the problem and its modeling in the form of the reinforcement learning model. Section 5 is about validity of proposed model (RLOM) by comparing it with two other known algorithms (1-1 algorithm and GA-based algorithm was proposed in 2002 by Kimbrough et al. [11]).

2. Literature review

Studies on supply chain inventory management generally recognize three stages, namely supply, production and distribution [3,5]. In a few cases, the researchers' focus is placed on the coordination and integration of inventory policies between more than three stages [11,16]. When there is no coordination among supply chain partners, each entity makes decision based on its own criteria, which results in local optimization as opposed to global optimum. Models for coordinated supply chain management are classified in three parts: buyer–vendor coordination, production–distribution coordination and inventory–distribution coordination. Firms have an opportunity to reduce operating costs while simultaneously improving customer service by coordinating the planning of these stages [31]. In the literature it is clearly shown that consideration of the entire supply chain including suppliers, manufacturers, distributors, and retailers, especially in cases with more than one actor in each level, is so complicated.

The bullwhip effect [13] is a critical issue in the supply chain management. As clarified in the literature [13,18], a small variance in the demands of the downstream customers may cause very high variance in the procurement quantity of upstream suppliers due to the bullwhip effect. The distortion of demand information can be viewed as a major factor in the formation of the bullwhip effect because of three related phenomena: (1) bias demand information from the downstream actors, (2) delay on information transferring between chain members, and (3) inappropriate logistical supports through the chain members [22].

In many researches, integration of all actors of supply chain is emphasized [5,10]. Simultaneous enhancement of efficiency and responsiveness needs coordination and integration of the whole supply chain partners. When the decision making processes of the supply chain partners are independent from each other, the received orders may not lead to a favorable supply policy for the upstream. The coordination of order and supply policies in two-echelon supply chain is investigated via bargaining models [26]. Also more strategies has been introduced in the literature for coordinating order and supply, one of most applicable is vendor managed inventory (VMI), in which, the retailers delegate the ordering and replenishment decisions to the manufacturer. Recently, integration of SC members under VMI initiative has been investigated by Yao et al. [33]. In a study on coordination among supply chain partners, the impact of order decision on reducing lead-times has been investigated in two-echelon supply chain. It is shown that, decreasing the order variability received by upstream, generates shorter and less variable lead-times. This introduces a compensating effect on the downstream inventory level by decreasing the order variability received by upstream [2].

Environment uncertainties intensify the need for coordination among chain members. The uncertainties can happen in various aspects. Main aspects of uncertainty investigated in literature, are demand uncertainty [5,6,8,10,11,21], and lead-times uncertainty [5,11,20]. Also beyond these two common types of uncertainties, some other types of uncertainties have been considered; e.g. impact of environmental uncertainties including customer uncertainty, supplier uncertainty, and technology uncertainty on information sharing and information quality has been considered [15]. An integrated system for managing inventories in a multi-echelon spare parts SC has been analyzed when chain involved very variable and lumpy demand; in which basic idea was separation of demand in two series (stable and irregular demand patterns) and adoption of proper forecast technique for each of them separately [10]. By capturing the trade-off between customer demand satisfaction and production costs, it has been shown service level can be improved for a reasonable increase in the total SC costs [8]. Concept of echelon stock for integrating inventory management in supply chain in a three levels SC has been considered in an environment with two uncertainty aspects: market demand and inventory holding and back-order costs [6]. Variability of lead-times between successive stages of the chain has a great effect on the coordination of supply chain. In one study [20] reducing both LT mean and variance in a two-echelon dual-sourced supply chain as an investment has been investigated. It has been shown coordinating the chain members in reduction of lead time reduce the total SC costs. One of the most crucial effects of demand uncertainties is the increasing inventory level and decreasing customer service, simultaneously [10]. Sheu addresses the issues regarding the uncertainty and complexity of the distortion of demand-related information existing broadly among supply chain members for efficient supply chain coordination [22].

Using inventory management policies such as order batching can distort the customer demand in upstream levels [14]. Ordering policies have a critical role in the inventory related costs and provided service level throughout the supply chain. In cases of deterministic demand with penalty cost (for unsatisfied orders) the optimal order for every member of the chain is the so-called “pass order” or “one for one” policy. According to 1-1 policy, each actor of chain orders to upstream whatever is ordered from downstream [11]. 1-1 policy is an ordering strategy appropriate for deterministic environments. Some ordering policies are introduced in literature in the uncertain environments [5,11]. Also, reinforcement learning model in three-level supply chain with

periodic inventory policy is applied by Giannoccaro and Pontrandolfo [5]. Although, there are some similarities between their work and this study in using reinforcement learning, nevertheless there are major differences such as action space, cost structure, and solving algorithm.

The Beer game [24] is a well-known example of supply chain which has attracted much attention from practitioners as well as academic researchers. Optimal parameters of the beer game ordering policy, when customers demand increases, have been analyzed in two different situations. It has been shown that minimum cost of the chain (under conditions of the beer game environment) is obtained when the players have different ordering policies rather than a single ordering policy [25]. Indeed, most of previous works on order policy of beer game use genetic algorithms as optimization technique [11,25]. But, in this study a reinforcement learning model is applied for determining beer game ordering policy. One ordering policy based on genetic algorithm under conditions of the Beer game environment was introduced [11]; we call that GA-based algorithm in this paper. GA-based algorithm has some degrees of freedom contrary to 1-1 algorithm; In the GA-based algorithm, each actor of chain can order based on its own rule and learns its own ordering policy in coordination with other members with the aim of minimizing inventory costs of the whole supply chain. One limitation of GA-based algorithm is the constraint of fixed ordering rule for each member through the time. We have addressed this limitation by the proposed model (RLOM).

3. Reinforcement learning model

Learning techniques are often divided into supervised, unsupervised and reinforcement learning (RL) methods. Supervised learning requires the explicit provision of input–output pairs and the task is constructing or mapping from one to the other. Unsupervised learning do not require target data, this method only performs processing on the input data. In contrast, RL uses a reward signal to evaluate input–output pairs and hence discover the optimal outputs for each input [23].

Reinforcement learning (RL) is learning what to do – how to map situations to actions – so as to maximize a numerical reward signal. The learner is not told which actions to perform in each situation, as in most forms of machine learning, but instead must find which actions yield the most reward by trying them in each state [27]. In another word, RL is the study of programs that improve their performance by receiving reward and punishments from the environment [28].

Basic idea of reinforcement learning is base on constant interaction between the learning agent and environment. The agent select an action and the environment respond to it and present a new situations to the agent [27]. Fig. 1 shows the agent–environment interaction in RL models.

As shown in Fig. 1 in the time step t agent takes action a_t based on the environment state. One time step later, in part as a consequence of its action, the agent receives a numerical reward r_{t+1} and find itself in the new state s_{t+1}. Reward r_{t+1} can be the criterion of selecting action a_t in state s_t but is not sufficient criterion because of the problem is long-term and rewards can only consider short-term consequences.

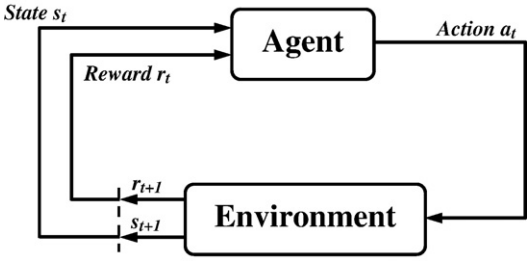

Fig. 1. Agent–environment interaction in RL models.

RL framework is simple and flexible thus it is possible to apply it to many different problems in many different ways [27]. In RL models the purpose or the goal of agent (or multi-agent system) is formalized in terms of a special reward signal passing from the environment to the agent (or multi-agent system) [27].

This learning method uses the agent's experience to improve the performance index with respect to a particular task [12]. Applications of RL methods are abound, mostly in the fields of game playing [29,30], robotics [19], scheduling [34] and inventory control [5,17].

Although the convergence property of RL has been widely investigated by machine learning researchers, but its applications to practical problems are still constrained by the curse of dimensionality [1].

3.1. Markov decision process

Many real-life decision making problems are found in stochastic environments, in these cases the uncertainty of environment state adds to the complexity of their analysis and create very complicated problems. A subset of these stochastic problems can be formulated as Markov or semi-Markov decision problems [7].

In the RL framework, the agent makes its decisions as a function of a signal from the environment called the environment state. A state signal that succeeds in holding all relevant decision making information is said to be Markov, or to have Markov property. Formal definition of Markov property is [27]:

$$Pr\{s_{t+1}=s',\quad r_{t+1}=r|s_t,a_t,r_t,s_{t-1},a_{t-1},r_{t-1},\ldots,s_0,a_0,r_0\}$$
$$=Pr\{s_{t+1}=s',\quad r_{t+1}=r|s_t,a_t\}. \qquad (1)$$

In another word, if the state signal has Markov property then the environment's response at time step $t+1$ depends only on the state and action representations at time step t, thus any relevant information for decision making is retained.

A reinforcement learning task satisfying the Markov property is called a Markov Decision Process, or MDP [27]. Standard RL theory has provided a comprehensive framework to solve Markov decision problems.

3.2. Parameters of RL model

In Markov environment, $P(s'|s,a)$ gives the probability of arriving state s' by selecting action a at state s [9]. States in

basic RL model should have Markov property but, even when the state is non-Markov, it is still appropriate to consider it as an approximation to a Markov state [27].

In reinforcement learning, rewards must represent the goals, which means, by maximizing the rewards, RL will improve the system toward its goals.

$Q(s,a)$ is defined as action-value function (or Q-function). The value function defines the summation of the discounted rewards accumulated toward the future [9].

If the $Q(s,a)$ was sufficiently large then selecting action a in state s is proposed by RL mechanism.

Definition of $Q(s,a)$ is:

$$Q(s,a) = E\{R_t | s_t = s, a_t = a\} \tag{2}$$

That:

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

Value of taking action a in state s is defined as the expected reward starting from s and taking the action a. γ is the discount rate and defined between 0 and 1.

It's possible to visualize Q-functions as a simple table of Q-values as in Fig. 2.

Solving a reinforcement learning task means finding a solution (a policy) that achieves as much reward as possible over the long run (i.e. value). Since, in most real world cases, complete and accurate model of the environment's dynamics is not achievable, calculation of future rewards is not possible and therefore $Q(s,a)$ must be estimated. Various learning methods have been developed for Q-function estimation (e.g. Q-learning [32]). After learning process is finished then, action with the highest Q-function is selected for each arriving state.

One of the challenges that appear in reinforcement learning and not in other kinds of learning is the trade-off between exploration and exploitation. The most important feature to distinguish RL from other types of learning is the use of training information to evaluate the actions taken rather than instructs by giving correct actions. This creates the need for proper exploratory behavior by explicit trial and error. On the other hand if RL maintains estimates of the action values, then at any given time at least one action exists whose estimated value is greatest; it is called greedy action. RL mechanism can exploit its current knowledge of the value of the action by selecting a greedy action (based on its criterion of best action). In the initial steps of learning the agents more explore, but in the next steps the probability of exploration decreases as the probability of exploitation increases. In each time step:

$$\Pr_{\text{exploration},t} + \Pr_{\text{exploitation},t} = 1. \tag{4}$$

Indeed, the agent in each time step has two choices: to explore (by probability $Pr_{\text{exploration,t}}$) or to exploit its knowledge (by probability $Pr_{\text{exploitation,t}}$). It is rational that agent mostly explores at first because of lack of knowledge about environment. In the next steps, contrary to the early steps, it is essential for agent to improve its knowledge (learning the value of Q-functions for each state–action pair) by testing the actions repeatedly in each state.

Policy in RL model is a mapping from each state s and action a to the probability of taking action a when the system is in state s. Optimal policy is the mapping from each state s to the best action on this state.

4. Problem description and modeling

A simple linear (vs. network) supply chain is considered in this paper which consists of four levels, namely supplier, manufacturer, distributor, and retailer, with only one actor at each level. Fig. 3 shows the supply chain model and its parameters.

According to the supply chain model shown in Fig. 3 three groups of variables are specified to characterize the supply chain:

$S_i(t)$ represents inventory position of level i in the time step t, $\quad i = 1,2,3,4$

$O_{ij}(t)$ represents ordering size of level i to the upstream level j, $\quad i = 0,1,2,3,4; \quad j = i+1$

$T_{ij}(t)$ represents distribution amount of level i to the downstream level $j, i = 1,2,3,4,5; j = i-1$.

As shown in Fig. 3 each actor distributes goods and services to the downstream actor. In this model O_{ij} and T_{ij} respectively represents information and material flows between sequenced supply chain actors. In this supply chain, customer demands must be met by retailer immediately otherwise the order is backlogged and retailer incurs the penalty cost. Also backlogged orders in each level are liable to penalty cost. The orders just received plus any backlog orders are filled if possible; the actor decides how much to order to replenish stock. In every level at each time step, four events happen: 1—previous orders are received (according to the lead-times) from upstream actor 2—order is received from downstream level 3—the received order is fulfilled from on-hand inventory (if possible) 4—actor decides about placing order for stock replenishment. This inventory system can be viewed as a periodic inventory review with one period cycle time (i.e. in each period, actor can place replenishment orders to the upstream after fulfillment of the downstream order).

We defined LT_{ij} as lead time of level i to the level j (that $j=i-1$) of the supply chain. As noted above $LT_{10}(t)=0$.

Order size of customer in this chain is uncertain i.e. no actor of chain knows new demand of customer before customer ordering. If retailer has sufficient stock then comply the customer demands, otherwise the backlog order cause penalty costs scaled to amount of backlogged orders.

	s_1	s_2	...	s_m
a_1	$Q(s_1,a_1)$	$Q(s_2,a_1)$	...	$Q(s_m,a_1)$
a_2	$Q(s_1,a_2)$	$Q(s_2,a_2)$	...	$Q(s_m,a_2)$
...	...	...	...	...
a_n	$Q(s_1,a_n)$	$Q(s_2,a_n)$	...	$Q(s_m,a_n)$

Fig. 2. Q-table.

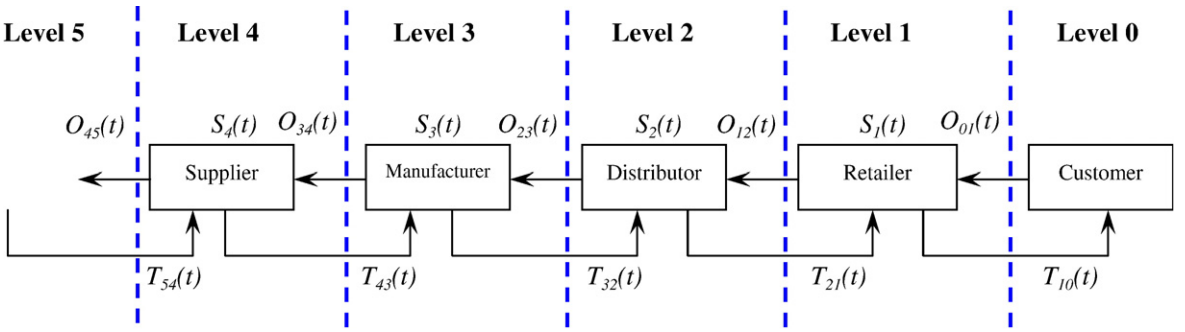

Fig. 3. Supply chain model.

Moreover in this problem we are faced with the uncertainties of lead-times. Lead time of any level of the chain except $LT_{10}(t)$ in the time step t is uncertain ($LT_{10}(t)=0$). It means that after receiving the order from downstream level to the actor of level i, if stock of level i was sufficient to comply the order then order is fulfilled but – by reasons of shipment problems, logistics uncertainties and so on – there is an uncertain lag between fulfillment of the order by actor i and its receipt by the actor $j=i-1$. Thus in each time step there are two uncertain parameters: 1—customer demand 2—lead-times (except consumer lead time).

The objective of supply chain ordering management is to determine quantity of O_{ij} in the way that total inventory cost of the chain consist of inventory holding cost and penalty cost of backlog orders is minimized:

$$\text{Minimize} \sum_{t=1}^{n} \sum_{i=1}^{4} [\alpha h_i(t) + \beta C_i(t)] \tag{5}$$

Where:

$$h_i(t) = \begin{cases} S_i(t) & \text{if } S_i(t)>0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

$$C_i(t) = \begin{cases} |S_i(t)| & \text{if } S_i(t)\leq 0 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

$h_i(t)$ is defined as on-hand inventory of level i at time step t and $C_i(t)$ is defined as backlog in level i at time step t. The latter is liable for the penalty cost. α and β are defined as inventory holding cost of each actor, per unit per period (e.g., in the MIT Beer Game, US$2 per case per week) and backorder cost of each actor/unit/period (e.g., in the MIT Beer Game, US$2/case/week), respectively[11]. n is time horizon and in our case is equal to 35 weeks. According to Eqs. (6) and (7), $h_i(t)$ and $C_i(t)$ are functions of inventory position on level i at time step t. Inventory position of each supply chain level at each time step is determined based on receiving orders from upstream (according to uncertain lead-times) and distributed amount to the downstream actor at time step t. On the other hand, both received orders from upstream and distributed amount to downstream, are functions of order size O_{ij} and therefore decision variables O_{ij} are embedded in the objective function 5.

4.1. Agent-based modeling of supply chain ordering system

To apply RL mechanism on the ordering management problem, described in this paper, it is necessary to formulate the problem in the form of RL models. As noted above RL models are applied in the agent-based framework, thus in the first step it is essential to model the ordering problem to the agent-based framework. In the next step characteristics of RL problem are defined on the designed agent-based framework.

Supply chain has various operations, each of which must be managed in the right manner. By agent-based modeling, each process (e.g. ordering) through the supply chain is considered as a multi-agent system. In the real world, each actor of the supply chain makes its own decisions autonomously about ordering size but in the case of minimizing the inventory cost of entire supply chain, it is essential for actors to cooperate, coordinate, and interact with each other. Fig. 4 shows the agent-based framework of supply chain ordering management system.

As shown in Fig. 4, to create an agent-based platform to apply RL mechanism on the ordering management problem, we consider SCOM as a multi-agent system that each ordering agent in this system is responsible for making decisions autonomously about ordering size by cooperating and interacting with other agents. Aim of this system is to achieve the common goal of minimizing the entire supply chain inventory cost. Note that in this type of modeling SCOM is one of SCM subsystems along with other subsystems such as transportation system, financial system, marketing system, and so on, each of them can be modeled as a multi-agent system.

Different activity must be done in SCOM such as negotiation mechanisms, controlling the system, learning mechanism, and so on. In our model there are four ordering agents in SCOM instead of four actors of the supply chain. The ordering agent of each echelon is identical to the other echelon's ordering agents. Every four agents must make their own decisions simultaneously and inform the ordering size to the upstream actor. Information on inventory positions is shared by ordering agents which is presented as system state in the format of a four element vector. Ordering agents through the

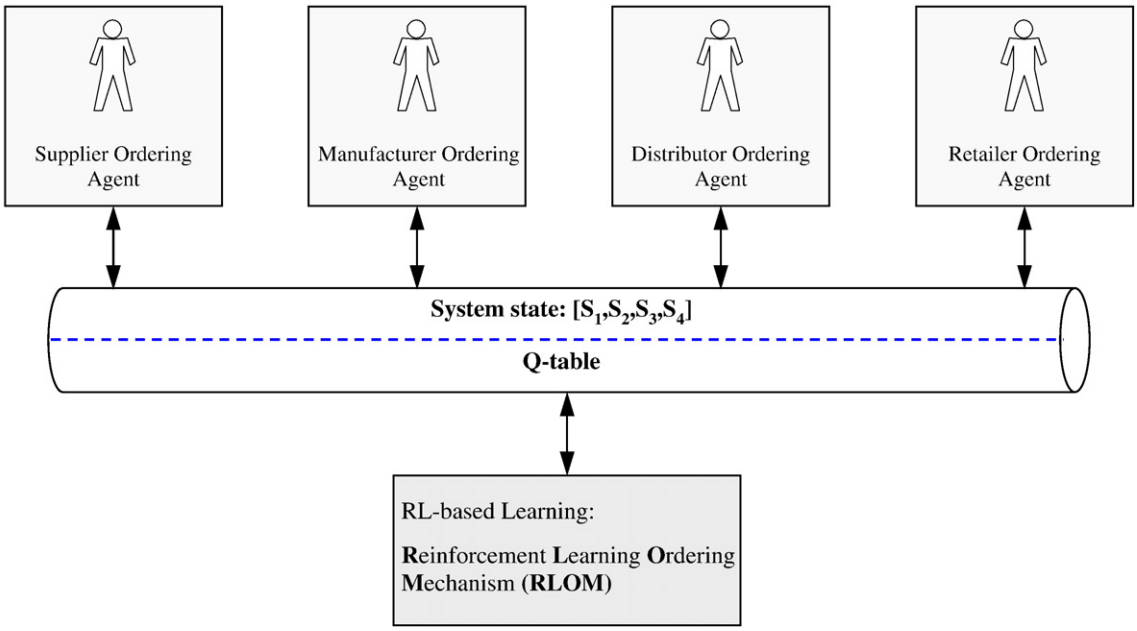

Fig. 4. Agent-based framework of supply chain ordering management system.

supply chain, coordinate with each other by meeting the system state and making decision through the common learning mechanism. In the next section this system has been modeled as a reinforcement learning problem.

4.2. RL modeling of ordering problem in the supply chain

In this section, characteristics of reinforcement learning model in the SCOM problem are defined and key parameters of RL model including the state variable, reward function, value function, and system policy are specified. Elements of reinforcement learning ordering mechanism (RLOM) are described here and the mechanism based on *Q*-learning is applied to solve the supply chain ordering problem.

4.2.1. State variable

As noted in previous section, if system state has the Markov property, then its one-step dynamics (see Eq. (1)) makes it possible to predict the next state and expected next reward given the current state and action. Nonetheless, while the state signal is not absolutely Markov, it is better to assume that it is approximation of Markov. In this way formulating the problem as a RL problem is possible. Since the final decision in RL models has been made according to the system state, it is essential for system state to provide proper information for agents' decision making process. In the SCOM problem we define the state vector as:

$$S(t) = [S_1(t), S_2(t), S_3(t), S_4(t)] \tag{8}$$

Where $S(t)$ is the system state vector at time step t and each $S_i(t)$ is the inventory position of the actor i at time step t. Thus at time step t a vector include four elements representing the system state that element kth stands for inventory position of actor k. This type of system state definition has been used in the literature [5]. It is clear that each element of state vector is infinite, thus determining the near-optimal policy is impossible because it needs infinite search power. For this reason we code the state variable to the finite set by mapping the state vector components to the finite numbers. Simulation showed that a coding strategy with 9 set is proper for our case. Coding strategy has been showed in Table 1.

It is clear that we mapped the infinite state size to the 6561 state.

4.2.2. Reward function

Objective of SCOM problem is the minimization of the total inventory costs thus we define the reward function as:

$$r(t) = \sum_{i=1}^{4} [h_i(t) + 2.C_i(t)] \tag{9}$$

Where $r(t)$ is the reward function at time step t that is a function of holding inventory size and shortage size at time step t. Since we applied our model to the MIT beer game problem, inventory holding cost of each actor per unit per period (coefficient of $h_i(t)$) and backorder cost of each actor/unit/period (coefficient of $C_i(t)$) must be set to 1 and 2, respectively. In this case, it is better to change the reward function to the loss function as we try to minimize it in the long term. It's clear that in each state we can simply calculate the loss function.

Table 1
Coding of the system state

Actual S_i	[− ∞;− 6[	[−6;−3[	[− 3;0[	[0;3[	[3;6[	[6;10[	[10;15[	[15;20[	[20;∞]
Coded S_i	1	2	3	4	5	6	7	8	9

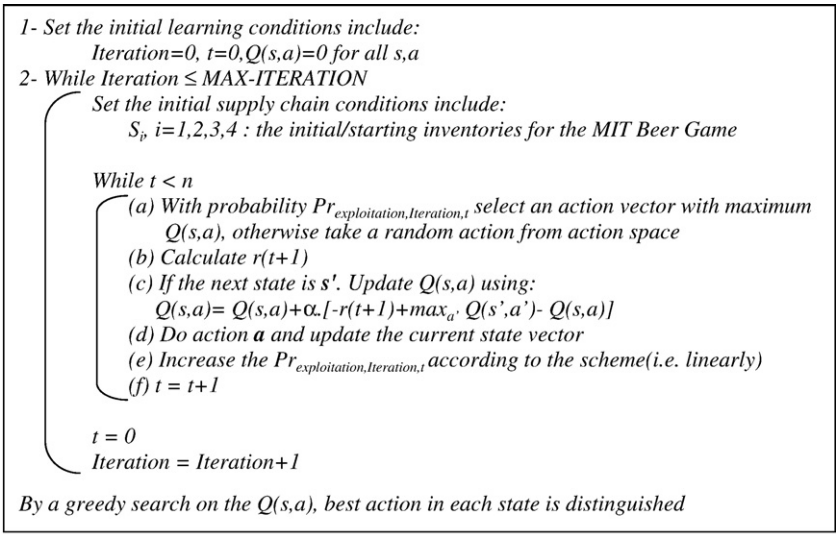

Fig. 5. Proposed algorithm based on *Q*-learning for solving SCOM problem.

4.2.3. Value function

Calculating the value function is not as simple as reward function because the next system states are uncertain as uncertainty is embedded in customer demand and lead-times. The customized version of value function applied in the RLOM is similar to general definition of $Q(s,a)$ (see Eq. (2)) in which:

$$R_t = \sum_{k=0}^{34-t} \gamma^k \sum_{i=1}^{4} [h_i(t+k+1) + 2 \cdot C_i(t+k+1)]. \quad (10)$$

Since discounting is not considered in RLOM, the discount factor γ in our model is chosen to be equal to 1.

Note that h_i and C_i for future periods cannot be calculated and therefore value of $Q(s,a)$ must be estimated. By estimating $Q(s,a)$ for each state–action pair, the best action in each state will be specified and optimal policy can be derived. In our model, *Q*-functions are estimated by a mechanism based on *Q*-learning. It's described in the next section.

4.2.4. (X + Y) ordering rule

Ordering rule applied in the RLOM named *X+Y* [11]. According to this rule, if the agent in the current period has *X* unit demand from the downstream actor, it orders *X+Y* unit to the upstream actor. *Y* can be zero, negative or positive i.e. the agent can order equal, greater or less than received order. *Y* is determined by learning mechanism. Simulation shows that the range [0,3] is suitable for *Y*s in our case. Action vector in period *t* is defined as $[Y_{1t},Y_{2t},Y_{3t},Y_{4t}]$, in which, Y_{it} denotes value of *Y* in *X+Y* rule for actor *i* in the time step *t*.

4.2.5. Agent's policy

In the RLOM, optimal agents policy is determined according to the learned value of $Q(s,a)$ i.e. for each state *s* action with greatest $Q(s,a)$ is selected. Thus after estimation of $Q(s,a)$ for all combination of *s*,*a* a greedy search on the *Q*-values can converge the algorithm to the near-optimal policy. In the RLOM, policy can be showed as:

$$Y_S = [Y_{1S}, Y_{2S}, Y_{3S}, Y_{4S}] \quad (11)$$

Where Y_S is the policy in the state *S*. Y_{iS} is the value of *Y* in the *X+Y* rule for actor *i* in the system state *S*.

To solve the problem it is necessary to determine agent polices according to the *Q*-values thus it is inevitable to estimate the *Q*-values. In the rest of this section the proposed algorithm to solve the SCOM problem is presented. This algorithm solves the reinforcement learning model of supply chain ordering problem.

Table 2
Four test problem data: include main test problem [4] and three new generated test problem

Experiment	Variable	Data (35 weeks)
Main test problem	Customer demand	[15,10,8,14,9,3,13,2,13,11,3,4,6,11,15,12,15,4,12,3,13,10,15,15,3,11,1,13,10,10,0,0,8,0,14]
	Lead-times	[2,0,2,4,4,4,0,2,4,1,1,0,0,1,1,0,1,1,2,1,1,1,4,2,2,1,4,3,4,1,4,0,3,3,4]
Test problem 1	Customer demand	[5,14,14,13,2,9,5,9,14,14,12,7,5,1,13,3,12,4,0,15,11,10,6,0,6,6,5,11,8,4,4,12,13,8,12]
	Lead-times	[2,0,2,4,4,4,0,2,4,1,1,0,0,1,1,0,1,1,2,1,1,1,4,2,2,1,4,3,4,1,4,0,3,3,4]
Test problem 2	Customer demand	[15,10,8,14,9,3,13,2,13,11,3,4,6,11,15,12,15,4,12,3,13,10,15,15,3,11,1,13,10,10,0,0,8,0,14]
	Lead-times	[4,2,2,0,2,2,1,1,3,0,0,3,3,3,4,1,1,1,3,0,4,2,3,4,1,3,3,3,0,3,4,3,3,0,3]
Test problem 3	Customer demand	[13,13,12,10,14,13,13,10,2,12,11,9,11,3,7,6,12,12,3,10,3,9,4,15,12,7,15,5,1,15,11,9,14,0,4]
	Lead-times	[4,2,2,0,2,2,1,1,3,0,0,3,3,3,4,1,1,1,3,0,4,2,3,4,1,3,3,3,0,3,4,3,3,0,3]

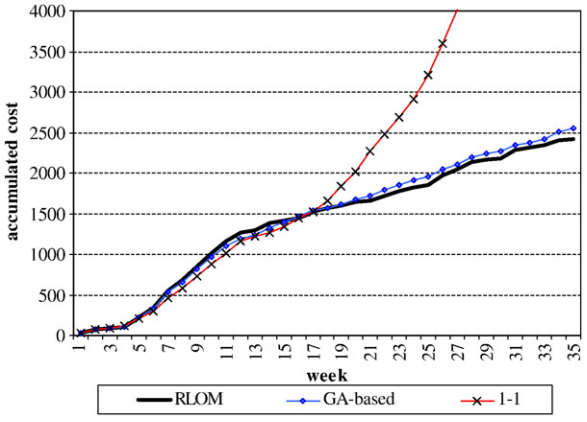

Fig. 6. Accumulated cost vs. week generated by RLOM, GA-based algorithm and 1-1 policy.

4.3. Proposed algorithm for solving RL ordering model

In previous sections problem is described and modeled in the form of reinforcement learning problem. In this part an algorithm for solving the RL modeled problem is proposed. Proposed algorithm is based on *Q*-learning mechanism that is a grand temporal difference method used for solving RL models [32]. In the proposed algorithm the value of *Q*-functions in the iterative process has been learned, in the end of learning process best action in each state is selected as optimal ordering policy to do for the same state in the future.

Fig. 5 shows the proposed *Q*-learning based algorithm.

As shown in Fig. 5 the *n*-periodic system simulated in the specific iterations (MAX-ITERATION) and the $Q(s,a)$ is learned during iterations. Indeed, in each iteration, supply chain ordering agents run for *n* periods. During the run time, inventory system works and in each state that system receives, one action is selected and $Q(s,a)$ is modified.

Probability of exploration is a function of iteration number and is reduced with increasing iteration number linearly (in our model from 98% in first iteration to 10% in last iteration). Also, in each specific iteration, it is reduced during period 1 to period *n* linearly (in our model from start probability – that is determined based on iteration number – to 2% at *n*th period).

Reward function during the simulation is calculated by Eq. (9). In each new state, calculation of reward function is straightforward. As shown in Fig. 5, it is possible to estimate the *Q*-function through the simulation by:

$$\begin{aligned} Q(s,a) &= (1-\alpha)Q(s,a) + \alpha \cdot [-r(t+1) + \max_{a'} Q(s',a')] \\ &= Q(s,a) + \alpha \cdot [-r(t+1) + \max_{a'} Q(s',a') - Q(s,a)] \end{aligned} \tag{12}$$

The rule (12) updates the state–action pair values. In the beginning, the agents have no knowledge about the value of each action in each state, thus the initial value of all *Q*-functions are set to zero for all state–action pairs. α is learning rate and must be defined between 0 and 1. Learning rate controls how much weight must be given to the reward just experienced, as opposite to the old *Q*-estimate. Note that a very small α can throwback the convergence of algorithm and a very large α (near to 1) can also intensify the effect of a biased sample and throwback the convergence of algorithm. Simulation shows that the learning rate 0.17 is suitable for our case.

The aim of ordering management system is to minimize the inventory cost of whole supply chain. Therefore, the reward function (9) must be minimized. In the learning phase, Eq. (12) is placed in the internal loop and *Q*-function is estimated for each state that agent gets to. By repeating each

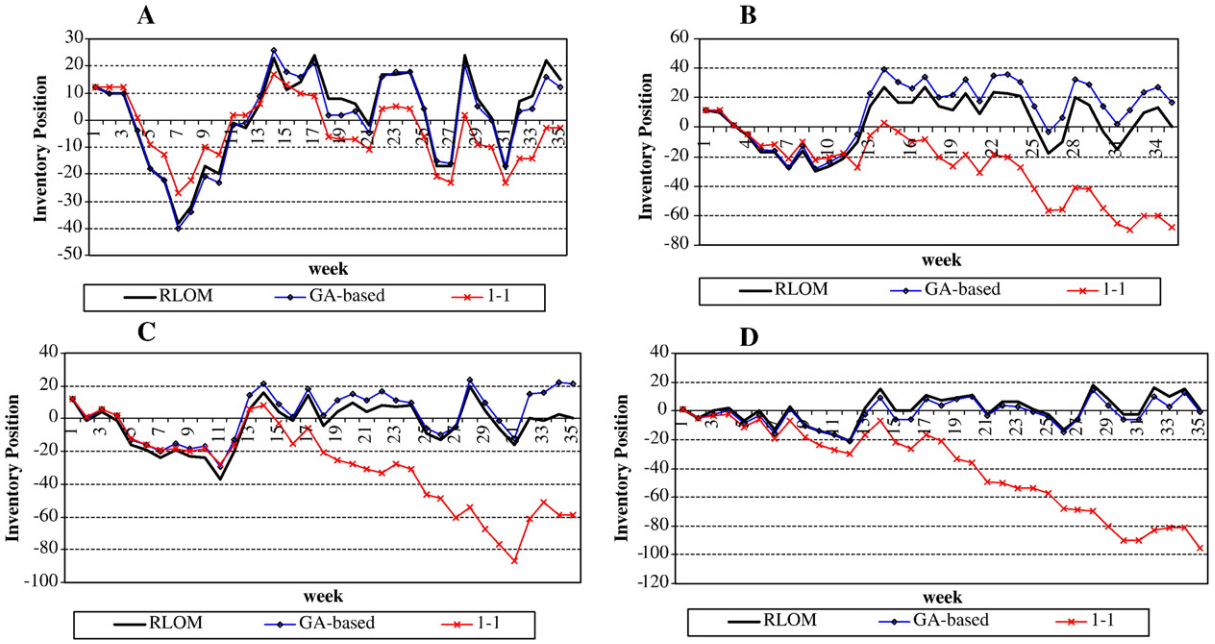

Fig. 7. Inventory position vs. week in each tree mechanism at each level of the chain (A: supplier inventory position B: manufacturer inventory position C: distributor inventory position D: retailer inventory position).

Table 3
Agent's policy in the RLOM (main test problem results)

Period	Retailer		Distributor		Manufacturer		Supplier		Cost
	IP*	Policy	IP*	Policy	IP*	Policy	IP*	Policy	
0	12	$x+2$	12	$x+2$	12	$x+2$	12	$x+1$	–
1	1	$x+3$	12	$x+1$	12	$x+3$	12	$x+3$	37
2	−5	$x+3$	−1	$x+0$	10	$x+2$	10	$x+0$	32
3	0	$x+1$	4	$x+0$	1	$x+0$	10	$x+2$	15
4	2	$x+1$	−1	$x+0$	−6	$x+2$	− 5	$x+0$	26
5	− 7	$x+2$	−16	$x+1$	− 17	$x+1$	− 18	$x+0$	116
6	0	$x+0$	− 19	$x+1$	− 17	$x+1$	− 22	$x+2$	116
7	− 13	$x+2$	− 24	$x+1$	− 28	$x+1$	− 38	$x+0$	206
8	3	$x+3$	− 19	$x+1$	− 16	$x+2$	− 32	$x+3$	137
9	− 10	$x+2$	− 23	$x+1$	− 30	$x+1$	− 17	$x+0$	160
10	− 14	$x+2$	− 24	$x+1$	− 26	$x+1$	− 20	$x+0$	168
11	− 17	$x+3$	− 37	$x+3$	− 21	$x+2$	− 1	$x+1$	152
12	− 21	$x+3$	− 20	$x+3$	− 10	$x+2$	− 3	$x+1$	108
13	2	$x+1$	6	$x+1$	14	$x+2$	6	$x+3$	28
14	15	$x+0$	16	$x+0$	27	$x+0$	23	$x+3$	81
15	0	$x+0$	4	$x+0$	17	$x+3$	11	$x+0$	32
16	0	$x+3$	− 1	$x+1$	17	$x+1$	14	$x+1$	33
17	11	$x+1$	14	$x+0$	27	$x+3$	24	$x+0$	76
18	7	$x+0$	− 4	$x+0$	14	$x+0$	8	$x+0$	37
19	9	$x+3$	4	$x+2$	12	$x+3$	8	$x+2$	33
20	11	$x+2$	10	$x+3$	23	$x+3$	6	$x+3$	50
21	− 2	$x+0$	4	$x+3$	9	$x+2$	− 2	$x+3$	21
22	6	$x+2$	8	$x+3$	24	$x+1$	17	$x+1$	55
23	6	$x+2$	7	$x+3$	23	$x+1$	17	$x+1$	53
24	1	$x+1$	8	$x+2$	21	$x+1$	18	$x+1$	48
25	− 2	$x+1$	− 9	$x+2$	1	$x+1$	4	$x+0$	27
26	− 13	$x+2$	− 13	$x+1$	− 18	$x+1$	− 17	$x+0$	122
27	− 6	$x+2$	− 5	$x+2$	− 10	$x+3$	− 17	$x+3$	76
28	18	$x+1$	20	$x+3$	20	$x+2$	24	$x+0$	82
29	8	$x+0$	5	$x+1$	15	$x+1$	8	$x+2$	36
30	− 2	$x+2$	− 6	$x+2$	− 3	$x+2$	1	$x+1$	23
31	− 2	$x+3$	− 16	$x+0$	− 15	$x+3$	− 18	$x+3$	102
32	16	$x+3$	0	$x+1$	− 3	$x+1$	7	$x+2$	29
33	10	$x+3$	− 1	$x+2$	10	$x+3$	9	$x+1$	31
34	15	$x+3$	3	$x+0$	13	$x+0$	22	$x+1$	53
35	1	–	0	–	0	–	15	–	16
Total cost=									2417

*IP means inventory position.

state through the simulation period, $Q(s,a)$ for each state–action pair converges. After finishing the learning phase, the best action in each state is retrievable by a greedy search on each state through the Q-table (see Fig. 2).

In the next section validity of proposed model is discovered by its compare with two other known mechanisms.

5. Experimental result and validity of RLOM

In this section validity of the proposed model is investigated. We compare our model with two other algorithms: 1-1 algorithm and GA-based algorithm [11]. Our test problem is based on beer game that is well-known problem in the supply chain research area. The initial/starting inventories for the MIT Beer Game are 12 cases of beer in the warehouse, 8 cases in the pipeline with 4 cases in the "truck" to be delivered to the warehouse in 1 week and 4 cases in the "train" to be delivered in the warehouse in 2 weeks [24] also game period is equal to 35 weeks. In the first experiment we use data from Kimbrough et al. [11]. In this test problem we are faced with both stochastic demand and stochastic lead-times, where demand is randomly generated from a known distribution, e.g., uniformly distributed between [0,15], and lead-times uniformly distributed from 0 to 4. Lead-times for all agents in the time step t are the same.

Also we generate three new test problems with parameters introduced by [11] (i.e. customer demand uniformly distributed between [0,15] and lead-times uniformly distributed from 0 to 4) and compare RLOM with two other algorithm. Table 2 shows the four test problems.

As noted above, in the first test problem we use data from [11]. In this case cost of whole supply chain inventory system achieved by RLOM is 2417. This is less than 2555 obtained using GA-based algorithm [11] and much less than 7463 obtained using the 1-1 policy. Fig. 6 shows accumulated cost vs. week in the each 3 algorithms.

Table 4
Comparison of inventory cost of RLOM with two other algorithms

	Main test problem	Test problem 1	Test problem 2	Test problem 3
1-1 policy	7463	5453	8397	7826
GA-based algorithm	2555	**3109**	4156	4330
RLOM	**2417**	3169	**4038**	**4205**

As shown in Fig. 6 accumulated cost in the RLOM seem to discover a dynamic order policy that outperforms GA-based algorithm and 1-1 policy. On the other hand it's profitable to know about inventory positions in each time step generated by each mechanism; Fig. 7 shows such inventory position by each mechanism.

As noted in the previous section our aim is to reduce inventory costs and it is achieved by supply chain inventory positions approaching to the zero in each moment, because there are no holding inventory cost and penalty cost in inventory position zero at each level of the chain. As shown in Fig. 7 inventory position of each level of the supply chain obtained using RLOM is less than the other two algorithms. Table 3 shows policy of supply chain agents proposed by RLOM in each time step of project. It is clear that agent's policy in each period is determined based on coded state of system e.g. in period 15 system state is [0,4,17,11] thus coded state (see Table 1) is [4,5,8,7], based on this coded state, proposed ordering policy by RLOM is [0,0,3,0].

We also tested for statistical validity by running the three new test problems. Table 4 shows comparison of RLOM with two other algorithms in four test problems (main test problem [11] and three new test problems).

As shown in Table 4, RLOM is much better than 1-1 policy in each of four test problems. Also RLOM is better than GA-based algorithm in three test problem. Only in one test problem GA-based algorithm is a bit better than RLOM.

1-1 policy is a static order policy that replenishes order in each period equal to order received from downstream (i.e. always Y is equal to zero). Algorithms such as 1-1 can discover optimal policy only in deterministic environments. GA-based policy has some dynamic aspects: it uses $X+Y$ policy with varying Y across agents but fixed over the time. RLOM as shown in Table 3, uses $X+Y$ rule and discovers dynamic ordering policy with varying Y across agents as well as over time. This two dimensional dynamics in the uncertain environment as we faced in this paper, is one of the reasons that RLOM outperforms two other algorithms. Ability of RL models in efficient search and use of the fact that the optimal or near-optimal policy is a function from state to action can be viewed as another advantage of the RLOM over two other algorithms.

6. Conclusion

In this paper the supply chain ordering management (SCOM) problem has been addressed. In the first step we proposed an agent-based supply chain ordering management in which agents manage ordering system of decentralized supply chain, in an integrated manner. In the next step we modeled SCOM as a reinforcement learning problem and in the final step the model was solved. Results show that proposed model (RLOM) is efficient and can find good policies under complex scenarios where analytical solutions are not available. This model is also adaptable to an ever-changing business environment.

In this problem we were faced with the case of stochastic demand and stochastic lead-times; in repeated simulations we paired the policy found by RLOM with 1-1 rules and GA-based algorithms [11]. We consistently found that RLOM performance is better than 1-1 and is almost better than GA-based algorithm (in three test problem RLOM found better solutions).

Our approach is based on three techniques, namely agent-based modeling, reinforcement learning and temporal difference methods for solving the RL problem. The approach has been tested on a linear supply chain model consisting of the Supplying, Manufacturing, Distributing and Retailing. The integrated ordering policy determined through the RLOM outperforms 1-1 policy and GA-based algorithm.

The summarized contributions of this paper are: (1) an agent-based supply chain ordering system is proposed. This framework is simple, flexible and aligned with process-based systems. (2) We design learning mechanism of supply chain ordering agents based on an interactive method. This learning mechanism lets agents to interact and be autonomous. Designing of learning model is completed in 2 stages: implementing reinforcement learning theory to the supply chain ordering problem and solving the derived model.

Results show that reinforcement learning is a powerful method to solve this problem. Furthermore potential of agent-based framework in supply chain management area is illustrated.

Further research should address the issue of having a non-linear (network) supply chain model. Furthermore, combining negotiation mechanism of agents, with better data sharing mechanism between agents and also combining the SCOM with other subsystems of SCM can be investigated in the future researches.

Acknowledgements

The authors wish to thank two anonymous referees for their helpful comments that enhanced the presentation of this paper.

References

[1] R.E. Bellman, Dynamic Programming, Princeton University Press, Princeton, 1957.
[2] R.N. Boute, S.M. Disney, M.R. Lambrecht, B.V. Houdt, An integrated production and inventory model to dampen upstream demand variability in the supply chain, European Journal of Operational Research 178 (2007) 121–142.
[3] S.S. Erenguc, N.C. Simpson, A.J. Vakharia, Integrated production/distribution planning in supply chains: an invited review, European Journal of Operational Research 115 (1999) 219–236.
[4] J.W. Forrester, Industrial Dynamics, MIT Press, Cambridge, MA, 1961.
[5] I. Giannoccaro, P. Pontrandolfo, Inventory management in supply chains: a reinforcement learning approach, International of Journal Production Economics 78 (2002) 153–161.
[6] I. Giannoccaro, P. Pontrandolfo, B. Scozzi, A fuzzy echelon approach for inventory management in supply chains, European Journal of Operational Research 149 (2003) 185–196.
[7] A. Gosavi, Reinforcement learning for long-run average cost, European Journal of Operational Research 155 (2004) 654–674.
[8] A. Gupta, C.D. Maranas, C.M. McDonald, Mid-term supply chain planning under demand uncertainty: customer demand satisfaction and inventory management, Computers and Chemical Engineering 24 (2000) 2613–2621.
[9] S. Ishii, W. Yoshida, J. Yoshimoto, Control of exploitation–exploration meta-parameter in reinforcement learning, Neural Networks 15 (2002) 665–687.
[10] M. Kalchschmidt, G. Zotteri, R. Verganti, Inventory management in a multi-echelon spare parts supply chain, International of Journal Production Economics 81–82 (2003) 397–413.
[11] S.O. Kimbrough, D.J. Wu, F. Zhong, Computers play the beer game: can artificial agents manage supply chains? Decision Support Systems 33 (2002) 323–333.
[12] I.S.K. Lee, H.Y.K. Lau, Adaptive state space partitioning for reinforcement learning, Engineering Applications of Artificial Intelligence 17 (2004) 577–588.
[13] H.T. Lee, J.C. Wu, A study on inventory replenishment policies in a two-echelon supply chain system, Computers & Industrial Engineering 51 (2006) 257–263.

[14] H. Lee, V. Padmanabhan, S. Whang, The Bullwhip Effect in Supply Chains, Sloan Management Review, 1997, pp. 93–102.
[15] S. Li, B. Lin, Accessing information sharing and information quality in supply chain management, Decision Support Systems 42 (2006) 1641–1656.
[16] W.Y. Liang, C.C. Huang, Agent-based demand forecast in multi-echelon supply chain, Decision Support Systems 42 (2006) 390–407.
[17] S. Mahadevan, N. Marchalleck, K.T. Das, A. Gosavi, Self-improving factory simulation using continuous-time average-reward reinforcement learning, Proceedings of the 14th International Conference on Machine Learning, 1997, pp. 202–210.
[18] R. Metters, Quantifying the bullwhip effect in supply chains, Journal of Operations Management 15 (2) (1997) 89–100.
[19] M. Riedmiller, Application of sequential reinforcement learning to control dynamic systems, Proceedings of 1996 IEEE International Conference on Neural Networks, 1996, pp. 167–172.
[20] S.W. Ryu, K.K. Lee, A stochastic inventory model of dual sourced supply chain with lead-time reduction, International Journal of Production Economics 81–82 (2003) 513–524.
[21] J.D. Schwartz, W. Wang, D.E. Rivera, Simulation-based optimization of process control policies for inventory management in supply chains, Automatica 42 (2006) 1311–1320.
[22] J.B. Sheu, A multi-layer demand-responsive logistics control methodology for alleviating the bullwhip effect of supply chains, European Journal of Operational Research 161 (2005) 797–811.
[23] A.J. Smith, Applications of the self-organising map to reinforcement learning, Neural Networks 15 (2002) 1107–1124.
[24] J. Sterman, Modeling managerial behavior: misperceptions of feedback in a dynamic decision making experiment, Management Science 35 (3) (1989) 321–339.
[25] F. Strozzi, J. Bosch, J.M. Zaldívar, Beer game order policy optimization under changing customer demand, Decision Support Systems 42 (2007) 2153–2163.
[26] E. Sucky, Inventory management in supply chains: a bargaining problem, International Journal of Production Economics 93–94 (2005) 253–262.
[27] R.S. Sutton, A.G. Barto, Reinforcement Learning: an Introduction, MIT Press, Cambridge, MA, 1998.
[28] P. Tadepalli, D. Ok, Model-based average reward reinforcement learning, Artificial Intelligence 100 (1998) 177–224.
[29] G.J. Tesauro, Practical issues in temporal difference learning, Machine Learning 8 (1992) 257–277.
[30] G.J. Tesauro, TD-Gammon, a self-teaching backgammon program, achieves master-level play, Neural Computation 6 (2) (1994) 215–219.
[31] D.J. Thomas, P.M. Griffin, Coordinated supply chain management, European Journal of Operational Research 94 (1) (1996) 1–15.
[32] C.J.C.H. Watkins, Learning from Delayed Rewards, PhD Thesis, University of Cambridge, England, (1989).
[33] Y. Yao, P.T. Evers, M.E. Dresner, Supply chain integration in vendor-managed inventory, Decision Support Systems 43 (2007) 663–674.
[34] W. Zhang, T.G. Dietterich, High-performance job-shop scheduling with a time-delay TD(l) network, in: D.S. Touretzky, M.C. Mozer, M.E. Hasselmo (Eds.), Advances in Neural Information Processing Systems 8: Proceedings of the 1995 Conference, 1996, pp. 1024–1030.

Dr. Kamal Chaharsooghi is Associate Professor of Industrial Engineering at the Dept. of I.E., Faculty of Engineering, Tarbiat Modares University, Tehran, Iran. Dr. Chaharsooghi's research interests include: manufacturing systems, supply chain management, information systems, strategic management, international marketing strategy and systems theory. Dr. Chaharsoogh's work has appeared in European Journal of Operational Research, International Journal of Advanced Manufacturing Technology, Scientia Iranica, Modares Journal of Engineering, Amirkabir Journal of Science and Technology, International Journal of Engineering Science. Dr. Chaharsooghi obtained his PhD from Hull University, England.

Jafar Heydari is a PhD candidate of Industrial Engineering in the School of Engineering at Tarbiat Modares University, Iran. He received his MSc from the same university in the Industrial Engineering and his BSc in industrial Engineering from Isfahan University of Technology. His main areas of research interests include agent-based modeling, learning models, supply chain management and meta-heuristics.

S.H. Zegordi is an Associate Professor of Industrial Engineering in the School of Engineering at Tarbiat Modares University, Iran. He received his PhD from Department of Industrial Engineering and management at Tokyo Institute of Technology, Japan in 1994. He holds an MSc in Industrial Engineering and Systems from Sharif University of Technology, Iran and a BSc in Industrial Engineering from Isfahan University of Technology, Iran. His main areas of teaching and research interests include production planning and scheduling, multi-objective optimization problems, meta-heuristics, quality management and productivity. He has published several articles in international conferences and academic journals including European Journal of Operational Research, International Journal of Production Research, Journal of Operational Research Society of Japan, Amirkabir Journal of Science and Engineering and Scientia Iranica International Journal of Science and Technology.