# Global Supply Chain Management: A Reinforcement Learning Approach

**Pierpaolo Pontrandolfo (pontrandolfo@poliba.it )[1]**

**Abhijit Gosavi (gosavi@uscolo.edu) [2]**

**[**]O. Geoffrey Okogbaa (okogbaa@ibl.usf.edu)[3]**

**Tapas K. Das (das@eng.usf.edu)[3]**

## Abstract

In recent years, researchers and practitioners alike have devoted a great deal of attention to supply chain management (SCM). The main focus of SCM is the need to integrate operations along the supply chain as part of an overall logistic support function. At the same time, the need for globalization requires that the solution of SCM problems be performed in an international context as part of what we refer to as Global Supply Chain Management (GSCM). In this paper we propose an approach to study GSCM problems using an artificial intelligence framework called reinforcement learning (RL). The RL framework allows the management of global supply chains under an integration perspective. The RL approach has remarkable similarities to that of an autonomous agent network (AAN); a similarity that we shall discuss. The RL approach is applied to a case example, namely a networked production system that spans several geographic areas and logistics stages. We discuss the results and provide guidelines and implications for practical applications.

[1.] Politecnico di Bari - Dipartimento di Progettazione e Produzione Industriale
Viale Japigia 182, 70126 Bari, ITALY

[2.] University of Southern Colorado, Industrial Engineering Program, 261, Technology Building,
2200, Bonforte Blvd, Pueblo, CO 81001 Phone: (719) 549-2788

[3.] University of South Florida,
College of Engineering
Department of Industrial & Management Systems Engineering
4202 E. Fowler Avenue, Tampa, FL 33647
(813)974-5576, Fax: (813)974-3651

[**] Corresponding Author

## 1. Introduction

According to Slats *et al*. 1995, logistics activities within an enterprise may be related to: (i) the feed-forward flow of goods; (ii) the feedback flow of information; (iii) management and control. Thus, logistics models a large part of a firm's activities that includes all the processes involved in delivering the final product to the consumer, namely supply, distribution and manufacturing/production. These processes span many of the business functions as well as different geographic areas. Often, manufacturing activities are dispersed and carried out in diverse locations with a focus on specific production phases or on some aspect of the final product.

The focus of this paper is on distributed systems. Such systems are characterized by facilities in dispersed locations, in which diverse or similar interdependent activities take place. The effective management of such systems is a key consideration in optimizing overall system performance. The diffusion of multi-plant production systems, market globalization, and the advances in the information and telecommunication technologies are among the most important reasons for the recent surge in interest in the operational issues and coordination of distributed production systems. Thus, there is a need to develop the research base and the analytical underpinnings of the key components of distributed production systems.

Multinational corporations (MNCs) are an example of distributed production systems with the added complexity of a supply chain that connects several nations with different currency, import tariffs, fiscal system, etc. In the last two decades there has been considerable increase in the number of coordination activities that have been initiated by the MNC’s based on industry specific factors (Flaherty, 1986). Several authors (e.g., Barlett and Ghoshal, 1993) analyzed integration issues under an organizational framework. A strategic perspective was adopted by Ferdows (1997) in looking at the coordination issues based on the strategic role of each foreign plant. Kogut (1990) addressed the issue of global coordination as a way to exploit network flexibility to achieve international sequential advantages. Doz (1986) looked at the importance of managing logistics so as to effectively implement an integration strategy.

This paper is concerned with coordination and integration of MNC's with emphasis on logistics and management of production processes. Logistics plays a key role in an international environment due to its impact on cost and time performances. While some work has been done with respect to the development of the conceptual frameworks for coordinating internationally dispersed manufacturing activities (Oliff *et al*., 1989), the literature contains very little by way of coordination at the operational level of a GSCM (Pontrandolfo and Okogbaa, 1998; Pontrandolfo, 1998). In particular, Vidal and Goetschalckx (1997) stated that there is very little research that addresses the integration of the single components of the overall production-distribution system, such as purchasing, production and scheduling, inventory, warehousing, transportation. Similarly, with respect to the coordinated planning between two or more stages of the supply chain, very little research currently exists that focuses on coordinating production, distribution, and scheduling (Thomas and Griffin, 1996). Today, several areas of GSCM are yet to be explored. As an example, import duties, which are generally 5–10% of the total product cost, need to be incorporated into the GSCM modeling framework. Also, the influence of a longer supply chain on the ability to quickly respond to consumer requirements merits further study.

To the best of our knowledge this work represents the first attempt to use a semi-Markov decision model on a problem related to GSCM. The application of Markov decision theory to SCM problems results in a large state-space. We work around this difficulty by using a reinforcement learning approach to solve the semi-Markov decision problem. Also, we analyze the coordination problem in networked production systems operated by MNCs and we develop methods to obtain solutions to operating strategies for MNCs. In the next section we propose the use of matrices to describe different approaches to the coordination of distributed systems as well as the characterization of GSCM. In section 3 we provide a brief introduction of semi-Markov decision processes (SMDPs) and reinforcement learning (RL), which we use in Section 4 to: (i) define a model of a specific multi-plant and multi-country production and distribution network, and (ii) determine the coordination policy of the system. Simulation results are discussed in Section 5. Section 6 contains the concluding remarks.

## 2. GSCM as an integrated approach to coordination

The available models dealing with logistics in distributed systems usually address specific aspects of the problem. Some models (e.g. Lootsma, 1994) are aimed at finding the optimal production allocation over the system life cycle in the long-medium term, i.e., production is assigned to the network facilities based on some optimizing criterion such as cost or time. Other models are concerned with coordinating or shifting production among facilities in the medium-short run as in Cohen *et al*. (1989) or Kogut and Kulatilaka (1994) respectively. Federgruen (1989) reviewed the planning models for the determination of integrated inventory and strategies in complex production/distribution systems, and reported that a single integrated, efficiently solvable model is still lacking.

To our knowledge, there is very little in the literature on how companies coordinate distributed systems. Arntzen *et al*. (1995) and Lee and Billington (1995) described how Digital Equipment Corporation and Hewlett-Packard dealt with this problem. The approaches used by both firms seem to be designed to manage their operations under an integrated perspective based on the GSCM concept. Logistics integration especially benefits those firms that are characterized by long supply channels and physical distances, which results in long pipelines and high throughput times. Integrating the supply chain allows information to be substituted for inventory in the channel (LaLonde and Masters, 1990). The resulting advantages are more meaningful where global sourcing or manufacturing is coupled with JIT sourcing (Fawcett and Birou, 1992) or lean production (Levy, 1997).

The coordination of distributed systems can be described in terms of (i) different functions, and (ii) different geographic areas. We refer to these as "function" and "area" respectively. We define a "function" as a specific logistics stage (e.g. distribution), whereas an "area" refers to a region (country or group of countries) that is homogeneous with respect to such variables as currency and import duties. We also consider that the lead times and the costs for the transportation between different areas are higher than within the same area. A distributed system involves several functions or several areas or both. The coordination priorities between the two dimensions need to be defined so as to describe how function or area interactions are handled.

Based on the foregoing, we propose a simple scheme (Figure 1) that can be useful in either classifying the existing models or in designing new ones from a particular perspective. Also, the scheme provides a specific view of the GSCM concept.

Three degrees of priority for coordination among the functions and the areas are considered within the scheme; namely, (i) no interaction (NO), (ii) low interaction (L), and (iii) high interaction (H). When interactions are considered at a low level, they are studied under an "interface" perspective, that is, functions or areas are expressed by suitable interfaces, for instance the inventory between manufacturing and distribution stages. Alternatively, interactions may be modeled under an "integration" perspective (high level), which would imply simultaneously coordinating manufacturing and distribution functions with the possibility of avoiding (or reducing) the intermediate inventory. This scenario results in a 3×3 matrix with 9 possible ways to coordinate systems that are multi-functional and covers multi-area. We do not consider the NO/NO case (gray background in Figure 1), given that in such a case the units of the distributed system would be completely independent.

——— Emphasis on: **production shifting** ———→

↑ Coordination strength along: **FUNCTION** |

| | NO | L | H |
|---|---|---|---|
| H | SCM | SCM + M-DM | GSCM |
| L | ECH | ECH + M-DM | ECH + GM |
| NO | | M-DM | GM |

↑ Emphasis on: **synchronizing and smoothing operations** |

——— Coordination strength along: **AREA** ———→

**Approaches for coordination**

| | | | |
|---|---|---|---|
| ECH: | echelon | SCM: | supply chain management |
| M-DM: | multi-domestic manufacturing | GM: | global manufacturing |
| GSCM: | global supply chain management | | |

*Figure 1. Approaches for coordination in distributed systems.*

Thus, four approaches can be defined, two for each dimension. The echelon (ECH) and the supply chain management (SCM) approaches respectively reflect the interface and the integration concepts for coordinating the different functions. Similarly, the multi-domestic manufacturing (M–DM) and the global manufacturing (GM) approaches refer to the coordination of different geographic areas under the interface and the integration perspectives respectively. By combining, four "hybrid" approaches result. In particular, the global supply chain management (GSCM) approach (H/H case) can be thought of as a combination of supply chain management and global manufacturing.

The shift from the interface to the integration in managing interactions involves considerable changes, which are key to understanding the basic features of the different approaches. Referring to Figure 1, we note the following.

1. As we move from L to H along “area” (shift from M–DM to GM), a growing emphasis is placed on the production shifting between facilities in the various geographic areas;
2. When moving from L to H along "function" (shift from ECH to SCM), a growing emphasis is placed on synchronizing and smoothing operations along the chain.

In the first case, the design and management of the distributed systems mainly address the optimization of manufacturing and transportation costs. In the second case, inventory costs as well as lead times become crucial. The GSCM approach emphasizes both shifting and smoothing and hence addresses manufacturing, transportation, and inventory costs simultaneously while trying to shorten lead times. In so doing, three key issues for logistics integration emerges, namely (i) the need for a systems approach aimed at a higher coordination along the supply chain as well as within every stage of the chain, (ii) the emphasis on logistic variables such as logistic costs, and (iii) the importance of a uniform production flow especially when time responsiveness is a crucial performance but long pipelines cannot be avoided, as is the case with global manufacturing in a JIT environment.

The GSCM approach aims at exploiting the advantages of distributed systems. Such advantages basically derive from higher flexibility associated with a larger number of options to accomplish a given task. However, to exploit such flexibility requires the ability to select the *best* option among *all* those available and this involves increased complexity in the decision-making process. According to Slats *et al*. (1995), the traditional operations research approach is well suited to the analysis of local performance of logistics sub-chains and processes, yet it is insufficient to fully support the performance analysis of an integrated logistics chain. In an attempt to overcome the drawbacks of the traditional operations research approach, we propose a methodology, based on artificial intelligence that permits the determination of an economic coordination policy for a GSCM.

In particular, we model the GSCM problem as a semi-Markov Decision Problem (SMDP) and solve it using the Reinforcement Learning (RL) algorithm. The fact that RL is a simulation based optimization approach makes it possible to consider detailed and more realistic descriptions of the problem.

## 3. Semi-Markov Decision Processes and Reinforcement Learning

A subset of problems in sequential decision-making domain under uncertainty can be modeled as Semi-Markov Decision Problems (SMDPs). Reinforcement Learning (RL) is a numerical method developed to solve these problems. RL's are especially useful when the size of these problems is large. First we briefly review the framework underlying an SMDP. Then we give a brief discussion of RL and its working mechanism. We conclude the section with a description of the RL algorithm that we used.

### 3.1. Underlying Framework

First we provide an intuitive idea of a Semi-Markov decision process and then give the mathematical definition and formulation. A Semi-Markov decision process is a stochastic process characterized by six elements (Puterman, 1994): decision epochs, states, actions, transition probabilities, transition rewards, and transition times. Also, there is a decision-maker that controls the path of the stochastic process. At certain points in time along the path, the decision-maker (agent) intervenes and makes decisions that affect the course of the future path. These points are called *decision epochs* and the decisions are called *actions*. At each decision epoch, the system occupies a so-called *decision-making state*. As a result of taking an action in a state, the decision-maker receives a reward (which may be positive or negative) and the system goes to the next state with a certain probability, called the *transition probability*. The amount of time spent in the transition (the *transition time) is a* random variable. A decision rule is used to select an action in each state while a policy is a collection of such decision rules over the state-space.

We now define the Semi-Markov decision processes mathematically. Let $X_m$ denote the state of the system at the $m$–th epoch and $T_m$ the time at the $m$–th epoch. Suppose, for each $m \in N$ (the set of integers), the random variable $X_m$ takes on values in a countable set $S$ and the random variable $T_m$ takes values in $\Re^+ = (0, \infty)$, such that $(0 = T_0 \leq T_1 \leq T_2 \leq \ldots.)$. For the stochastic process $(\mathbf{X}, \mathbf{T}) = \{X_m, T_m: m \in N\}$ with state space $S$, for all $m \in N$, $j \in S$, and $t \in \Re^+$, the following condition is satisfied:

$$P\{X_{m+1} = j, t \geq T_{m+1} - T_m \mid X_0, \ldots, X_m, T_0, \ldots, T_m\} = P\{X_{m+1} = j, t \geq T_{m+1} - T_m \mid X_m\}.$$

Hence, the process $(\mathbf{X}, \mathbf{T})$ is a Markov renewal process. Also assume that $S$ is finite and the Markov chain $\mathbf{X} = \{X_m: m \in N\}$ is irreducible. Define a process $\mathbf{Y} = \{Y_t: t \in \Re^+\}$, where $Y_t = X_m$, if $t = T_m$. Then $\mathbf{Y}$ is a Semi-Markov process associated with the process $(\mathbf{X}, \mathbf{T})$. It may be noted that the process $\mathbf{Y}$ changes continuously with time, however decisions are only made at specific system state-change epochs making it possible for the system to change its state several times between two decision epochs. The stochastic process $(\mathbf{X}, \mathbf{T})$ that is embedded in Y tracks the decision-making epochs of $\mathbf{Y}$. We call **(X, T)** the *decision process*, and the complete process $\mathbf{Y}$, the *natural process*. For $i \in S$, when action $a \in A_i$ is chosen (for any state $i$, $A_i$ denotes the set of possible actions that can be taken in $i$), and if the next state of the decision process is $j$, let $r(i, j, a)$ represent the immediate reward obtained and $t(i, j, a)$ represent the time spent, during the state-transition. Also let $i_k$ represent the state visited in the $k$–th decision epoch and $\mu_k$ represent the action taken in that epoch. Then the average reward (gain) of an SMDP starting at state $i$ and continuing with policy $\pi$ is given as:

$$g^{\pi}(i) = \frac{\lim_{N\rightarrow\infty}\left\{E\left[\sum_{k=1}^{N}(r(i_k, i_{k+1}, \mu_k \middle| i_0 = i_1)\right]\middle/ N\right\}}{\lim_{N\rightarrow\infty}\left\{E\left[\sum_{k=1}^{N}(t(i_k, i_{k+1}, \mu_k \middle| i_0 = i_1)\right]\middle/ N\right\}}.$$

In this paper, our focus is on problems with average reward performance measure over an infinite time horizon.

Value iteration and policy iteration are two popular dynamic programming (DP) approaches that are used in solving SMDPs. Please see Puterman (1994) for more detailed discussion on these methods. However both value and policy iteration algorithms suffer from the “curse of dimensionality” as well as the “curse of modeling” respectively, and are of limited use when the problem state space is large and the underlying model is complex.

### 3.2. Reinforcement Learning

RL is a relatively new simulation-based method that makes use of both value iteration and policy iteration. Thus an RL based approach is ideally suited for decision-making problems that are either Markov, or semi-Markov decision problems (MDPs or SMDPs). It does not suffer from some of the drawbacks of DP, and thus it can be used for problems with large state spaces. For more detailed discussion on RL, please refer to the texts by Sutton and Barto (1998) and Bertsekas and Tsitsiklis (1996). Next we briefly describe mechanism underlying value iteration based RL.

#### 3.2.1. Learning Model

An RL model consists of four elements namely, an environment, a learning agent with its knowledge base, a set of actions that the agent can choose from, and the response from the environment to the different actions in different states. The knowledge base is made up of the so-called *R-values* for each state-action pair. By examining these values (which may be in terms of costs or rewards) for the different actions that may be taken in the current state, the agent decides which action to take in that state. The response to every action is simulated. To be able to do this effectively requires complete knowledge of the random variables that govern the system dynamics.

The mode in which the learning occurs is as follows. Before the learning begins, the values *R*(*i*,*a*) for all states *i* and all actions *a* are set to the same value. When the system is in a decision-making state *i*, the learning agent examines the values for all actions in state *i* and selects the action *u* that maximizes the value (if values are in terms of reward). This action leads the system along a unique path until the system encounters another decision-making state (*j*). During the state-transition (from *i* to *j*), the agent gathers information from the environment about the immediate reward earned and the time spent during the state change. This information is used by the agent with the help of the learning algorithm to update the value *R*(*i*,*u*). A poor action results in a decrease in this value while a good action that results in high rewards increases the value (and *vice-versa* if the value is in terms of cost). The exact change in value is determined from the RL algorithm SMART (discussed in the next section). In other words, the performance of an action in a state is used to update the knowledge base. Thus every experience makes the agent a little more aware of the environment than before. Since all state-action pairs are encountered theoretically an infinite number of times, over time the agent learns the right actions to take in each state.

When the system visits a state, the agent selects the action with the highest (or lowest if it is in terms of costs) *R-value* for that state-action pair. Therefore in the event that an action produces a high immediate reward, the updating causes the agent to be partial to that action. Similarly, the opposite is possible when the action produces a low immediate reward. This can cause considerable difficulty for the agent with respect to learning the right policy because the short-term effect of an action can be misleading. Consequently it becomes necessary to try all actions in all states. Therefore occasionally the agent has to divert from the policy of selecting the most preferred action (greedy strategy) and instead selects some other action. This is called *exploration*. As good actions are rewarded and bad actions punished over time, some actions tend to be more and more preferable and others less so. The learning phase ends when a clear trend appears in the knowledge base and a deterministic policy is learned. Of course, by this point, exploration must cease.

### 3.2.2. SMART

SMART (Das *et al.*, 1999) is an acronym for Semi-Markov Average Reward Technique. It is a value iteration based learning algorithm for solving SMDPs under the average reward criterion over an infinite time horizon. The system response for each action is captured from simulating the system with different actions in all states. Information about the response is obtained from the immediate rewards received and the time spent in each transition from one decision-making state to another. Within the simulator, the optimum R-values are learned and with them a deterministic policy. The updating of R-values (learning) uses a so-called learning rate. The learning rate has to be gradually decayed to 0 as the learning progresses. The probability of exploration is also similarly decayed to 0. A typical decaying scheme may be given as follows: $a_m = M/m$ where $a_m$ is the value of the variable (learning rate or exploration probability) at the $m$ th iteration and $M$ is some predetermined constant. Typically $M$ is about 0.1 for exploration probability and 0.01 for learning rates. The SMART algorithm is given in Figure 2.

**SMART algorithm**

```
1.  Let time step $m = 0$. Initialize action values $R(i, a) = 0$ for all $i \in S$ and $a \in A_i$. Set the
    cumulative reward $C = 0$, the total time $T = 0$, and the reward rate $\rho = 0$. Start system
    simulation.
2.  While $m <$ MAX_STEPS do
    If the system state at the time step $m$ is $i \in S$,
    (a) Decay $\alpha_m$ and $p_m$ according to some scheme (as explained in Section 3.2.2).
    (b) With probability 1- $p_m$, choose an action $a \in A_i$ that maximizes $R(i, a)$, otherwise
        choose a random (exploratory) action from the set $\{A_i \setminus a\}$.
    (c) Simulate the chosen action. Let the system state at the next decision epoch be $j$. Also
        let $t(i, j, a)$ be the transition time, and $r(i, j, a)$ be the immediate reward earned as a
        result of taking action $a$ in state $i$.
    (d) Change $R(i, a)$ using:
    $R(i, a) \leftarrow (1-\alpha_m)R(i, a) + \alpha_m\{r(i, j, a) - \rho t(i, j, a) + \max_b R(j, b)\}$
    (e) In case a non-exploratory action was chosen in step 2(b)
        •  Update total reward $C \leftarrow C + r(i, j, a)$
        •  Update total time $T \leftarrow T + t(i, j, a)$
        •  Update average reward $\rho \leftarrow C/T$
        Else, go to step (f).
    (f) Set $i \leftarrow j$, and m $\leftarrow$ m+1.
```

*Figure 2. SMART: A RL algorithm for computing gain-optimal policies for SMDPs.*

## 4. Similarities and differences or RL with the framework of autonomous agents

The reinforcement learning (RL) framework bears remarkable similarities to the Autonomous Agent Network (AAN) framework (Huang and Nof, 2000). The philosophy of distributed problem solving (DPS) is employed in an AAN framework. It divides a large problem into smaller tasks and associates an agent with each task. The agents cooperate with each other through communication protocols. As a result, a complex decision-making problem is not decomposed into sub-problems, which could be solved independently. Rather a collection of agents now collaborates among themselves to tackle the problem as a group. In this respect, the RL framework has similar features as AAN, which we discuss next.

In an RL framework, we associate an agent with *each state*. The knowledge base of each agent contains a finite number of R-values; the number of values equaling the number of actions allowed in that state. The knowledge base gets continually updated in a simulator through the learning process, which is closely related to the semi-Markov decision process in the GSCM context. These agents frequently exchange information in the form of the current average reward (ρ) and some R-values. Dynamic programming principles, which form the basis of RL theory, dictate that this information be exchanged between agents to ensure that they behave in a cooperative manner. It may also be noted that the information exchanged in RL is thus compact and hence information overload is avoided in the communication.

A powerful feature of RL is that it can produce good team behavior in an uncertain environment. Uncertainty has been a recurrent issue in the agent literature (see Gasser, 1994, and Balakrishnan et al, 1994). Uncertainty renders the decision-making problem more complex and challenging and may actually limit the capabilities of autonomous agents. RL seem to be well suited to these challenges given the fact that stochastic dynamic programming and Monte Carlo simulation are the two major are the basic foundation of RL.

In any RL implementation of the GSC, the frequency of information exchange is dictated by the simulator and by the semi-Markov decision process. When learning occurs, the simulator is bombarded with messages from these agents more or less regularly. The messages are of the form “Yes” or “No;” if there are two actions to select for each agent. Of course, in case of multiple actions, the quality of messages will be different. The fact remains that these messages affect the nature of the trajectory of the semi-Markov process. In any large-scale implementation of RL, a difficulty that is commonly experienced that of “explosion in the number of agents.” This is often referred to as the “curse of dimensionality.” The major obstacle faced in many real-life implementation of RL is the large number of states encountered and consequently a large number of agents. A number of strategies have been explored in an attempt to circumvent this difficulty. A commonly used strategy is to integrate an Artificial Neural Network (ANN) within the RL simulator. The literature on agents (see Huang and Nof, 2000) discusses several strategies to “cluster” agents without losing their cooperative nature. This seems like an attractive possibility in RL and deserves further research attention.

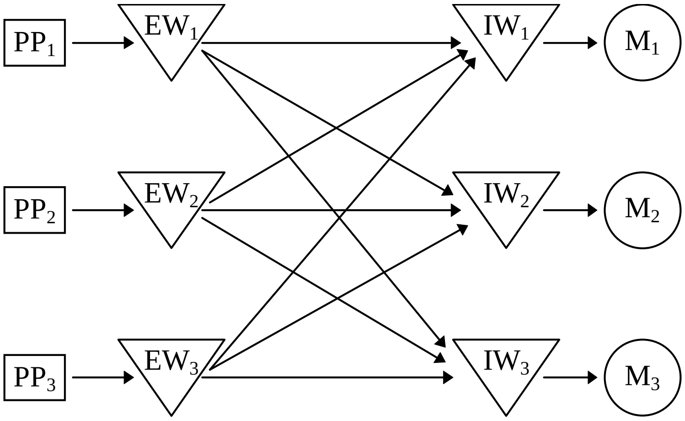
In summary, it can be said that RL has some striking similarities with AANs. The philosophy of cooperative agent behavior and information exchange seems to be a strong common thread tying both frameworks. RL differs in its implementation aspects; it has a dynamic programming and simulation backbone, which endows it with some superior features such as robust behavior in uncertain environments and compact information exchange laws, and also some weaknesses such as the “curse of dimensionality.” It is especially in this last aspect, namely the curse of dimensionality, that the clustering principles of autonomous agents should provide some reliable strategies for RL.

## 5. Coordinating distributed production systems through reinforcement learning

In the next two Sections, we describe how SMDPs and RL can help managers to coordinate distributed production systems under the GSCM concept. First we describe the system model, and then discuss the associated SMDP.

### 5.1. A model of a distributed production system

The model of a distributed production system that we consider in this work spans two functions, namely manufacturing and distribution. It also encompasses three geographic areas (Figure 3). The system manufactures and delivers a single product. Cost and time variables and their respective values (adopted somewhat arbitrarily) for a numerical evaluation of the proposed approach are shown in Table 1. Notice that any difference among countries is modeled through the appropriate real currency exchange rate: for example, the actual manufacturing cost is the product of the exchange rates and a value that does not change with country. Tariffs are added on transportation costs when inter-country shipments occur. A penalty cost is applied for late deliveries, namely when the product is not available in stock to fulfill market demand. The penalty cost accounts for the system responsiveness, which often prove to be more critical than efficiency, especially when delivering *innovative* products. In fact, the key logistics function for these products is *market mediation* rather than the *physical* function. That is, the cost associated with managing the interface between production and market is more important than the cost associated with transportation and inventory (Fisher, 1997).



$PP_i$ = production plant in the *i* th country
$EW_i$ = export warehouse in the *i* th country
$IW_i$ = import warehouse in the *i* th country
$M_i$ = final market in the *i* th country
*i* = country index (*i* = 1, 2, 3)

*Figure 3. The model of the production/distribution network.*

Based on Table 1, the reward ($r$) obtained from every product manufactured in the *i* th country and delivered to the *j* th country can be determined as follows:

$$r = e_j \cdot p - \left(e_i \cdot m + I \cdot e_i \cdot it_i + t_{ij} + c \cdot tt_{ij} + I \cdot e_j \cdot it_j\right) - \mathrm{IJ} \cdot d_j \cdot e_j \cdot p - \mathrm{LATE} \cdot e_j \cdot l,$$

| | | |
|---|---|---|
| **Costs** | | |
| Real currency exchange rate (uniform distribution) | | |
| $i = 1$ | $e_1$ | = 0.4 to 0.6 |
| $i = 2$ | $e_2$ | = 0.9 to 1.1 |
| $i = 3$ | $e_3$ | = 1.9 to 2.1 |
| Unit price | $p$ | = 500 |
| Unit production cost | $m$ | = 300 |
| Unit transportation cost | | |
| within country ($i = j$) | $t_{ii}$ | = 30 |
| between countries ($i \neq j$), slow carriers | $t_{ij}$ | = 45 |
| between countries ($i \neq j$), fast carriers | $t_{ij}$ | = 60 |
| Tariff (for inter-country shipment) | $d_i$ | = 15% |
| Unit inventory cost (per time unit) | $I$ | = 10 |
| Unit pipeline cost (per time unit) | $c$ | = 6 |
| Penalty cost for late delivery | $l$ | = 3 |
| **Times** | | |
| Production time (uniform distribution) | $mt$ | = 1 to 3 |
| Transportation time (uniform distribution) | | |
| within country ($i = j$) | $tt_{ii}$ | = 1 to 3 |
| between countries ($i \neq j$), slow carriers | $tt_{ij}$ | = 15 to 17 |
| between countries ($i \neq j$), fast carriers | $tt_{ij}$ | = 3 to 5 |
| Time in inventory in the $i$ th country | $it_i$ | |
| Demand rate (time between demands: erlang distribution) | | |
| $i = 1$ | $\lambda_1$ | = 0.1715 |
| $i = 2$ | $\lambda_2$ | = 0.3430 |
| $i = 3$ | $\lambda_3$ | = 0.6860 |

*Table 1. Cost and time variables.*

where the first element on the right hand side represents the revenue earned, the elements within the parentheses are: production cost, inventory carrying cost in country *i*, transportation cost from country *i*, country *j*, inventory carrying cost during transportation from country *i* to county *j*, inventory carrying cost in country *j* respectively. The last two elements denote the tariff cost and the penalty cost respectively paid in country *j*. The variables IJ and LATE are binary logic variables defined as follows:

$$\text{IJ} = \begin{cases} 1 \text{ if } i \neq j \\ 0 \text{ if } i = j \end{cases} \text{ and LATE} = \begin{cases} 1 \text{if the delivery is late} \\ 0 \text{ otherwise.} \end{cases}$$

Notice from Table 1 that the local demand rates are different since they have been assumed proportional to the local exchange rates. The demand rates are also set such that the average total demand rate of 1.2 per unit time is equal to 80% of the total production rate of 1.5 (combined rate of 3 countries, where the mean production time of each product is 2.0 time units giving a rate of 0.5 per country). Therefore, foreign production allows the system to better balance production with demand as well as to exploit differences in the exchange rates. This requires that the system be managed under an integration perspective according to a coordination policy that is economical over the long run with respect to the total reward.

In order to better coordinate the system under an integration perspective, interactions among both functions and areas has been modeled coherently with the GSCM concept. For example, under logic of integration along the "function" dimension, decisions are made on the basis of the performance of the whole system, rather than on the performance (e.g. minimum fill rate) required by market to distribution and in turn by distribution to manufacturing. Similarly, under logic of integration along the "area" dimension, production is assigned to the manufacturing facilities based on cost variables that includes the values of currency exchange rates and tariffs for inter-country shipments.

Econometrics of the coordination policy over the long run requires that this be determined so as to maximize the sum of the unit rewards ($r$) over an infinite horizon. This would be quite difficult in practice and hence as discussed in Section 3.1, we maximize the average reward (gain) $\rho$. Optimality of the policy so determined is guaranteed, provided such a policy is repetitive after an adequate warm-up period.

According to the GSCM concept, the backward flow of orders (from markets to production plants) is assumed to be instantaneous, namely as soon as a market demand arises, a production order is issued to one of the production plants. In particular, when demand occurs in a certain $\text{M}_j$, a withdrawal order is issued to the IW$_j$ in the same country and, if it is available, a product is withdrawn and delivered to the customer. If the product is not available, the demand is backordered. The demand is disposed of if the maximum backorder level is reached. At the same time, unless the demand has been disposed, a withdrawal order is issued to one of the EW$_i$, according to a decision rule. The decision rule is to select an appropriate EW$_i$ as well as the choice of the transportation mode. Once the order has been issued to an EW$_i$, either a product is immediately withdrawn (if it is available) or the order is backordered (no maximum limit exists for backorder at all EW$_i$). As soon as an EW$_i$ receives a withdrawal order, a production order is issued to the PP$_i$ in the same country to replenish the inventory. One key assumption for the coordination policy concerns the decision rule utilized to allocate the withdrawn orders to the EW$_i$: only stationary decision rules are considered, namely decisions that solely depend on the system state and not on time.

### 5.2. SMDP Model

The choice of a semi-Markov model as opposed to a Markov model is dictated by the fact that the performance metric in the GSCM context is more likely to be calculated over a time frame (an infinite or finite time horizon). The semi-Markov model permits such implementation. The problem of determining an economic coordination policy for the system described in the foregoing paragraphs is a decision problem in which every time a new demand arises in a local market, the decision agents must make a decision solely based on the system state. Hence, the system state has to be defined so as to provide relevant information to the agents. Furthermore, given the decisions that can be made for three different countries, i.e. IW$_1$, IW$_2$ and IW$_3$ there must exist three decision agents. Every decision agent must select one of the following options: allocating the withdrawal order to replenish inventory at the IW$_i$ either to the local export warehouse or to one of the other two export warehouses; in the later case the agent can select between two transportation modes, namely the slow and the fast carriers. This results in five available actions in every state and for each agent.

It has been assumed that the key information necessary to make decision concerns the order status at each warehouse, which depends on the warehouse size as well as on the current inventory and the backorder status. In particular, every warehouse is characterized by the number of unfulfilled orders as shown in Table 2.

| | |
|---|---|
| Maximum stock at $\mathrm{EW}_i$ | $\mathrm{EWS}_i = 2$ |
| Current inventory at $\mathrm{EW}_i$ | $\mathrm{EWI}_i = 0, 1, 2$ |
| Maximum backorder at $\mathrm{EW}_i$ | $\mathrm{EWB}_i = +\infty$ |
| Number of unfulfilled orders at $\mathrm{EW}_i$ ($=\mathrm{EWS}_i-\mathrm{EWI}_i$) | $\mathbf{EWN}_i = 0, 1, 2$ |
| Maximum stock at $\mathrm{IW}_i$ | $\mathrm{IWS}_i = 2$ |
| Current inventory at $\mathrm{IW}_i$ | $\mathrm{IWI}_i = 0, 1, 2$ |
| Maximum backorder at $\mathrm{IW}_i$ | $\mathrm{IWB}_i = 2$ |
| Number of unfulfilled orders at $\mathrm{IW}_i$ ($=\mathrm{IWS}_i-\mathrm{IWI}_i+\mathrm{IWB}_i$) | $\mathbf{IWN}_i = 0, 1, 2, 3, 4$ |

**System state** = ($\mathrm{EWN}_1$, $\mathrm{EWN}_2$, $\mathrm{EWN}_3$, $\mathrm{IWN}_1$, $\mathrm{IWN}_2$, $\mathrm{IWN}_3$)

*Table 2. System state variables.*

It may be noticed that the correct expression for the number of unfulfilled orders at any export warehouse should be:

$$\mathrm{EW}_i = \mathrm{EWS}_i-\mathrm{EWI}_i+\mathrm{EWB}_{i,}$$

which is similar to the expression used for import warehouses. However, given that no maximum limit exists for backorders at all export warehouses, this would involve an infinite number of system states. To reduce the state space, we decided to collapse all the actual values of $\mathrm{EW}_i$ ranging from 3 to $+\infty$ into $\mathrm{EW}_i = 2$. As a result, the decision agent considered the case of an empty stock in the same manner as the case of an empty stock with backorder. The system state is given by the following six-component vector:

$$(\mathrm{EWN}_1, \mathrm{EWN}_2, \mathrm{EWN}_3, \mathrm{IWN}_1, \mathrm{IWN}_2, \mathrm{IWN}_3),$$

whose values range from (0, 0, 0, 0, 0, 0) to (2, 2, 2, 4, 4, 4). Thus, every decision agent has to explore a state space of $3^3 \times 5^3 = 3375$ states and find the best action in each of the states; namely, selecting one of three possible suppliers and choosing one of two transportation modes, for any inter-country shipment. This results in 16875 action values to define the policy of a single agent and 50625 to define the global policy of the three agents. Notice that the agents make decisions based on the same information, namely a unique system state variable, and share the reward function. One might argue that the *i* th agent could decide based on a simpler state variable, for instance:

$$(\mathrm{EWN}_1, \mathrm{EWN}_2, \mathrm{EWN}_3, \mathrm{IWN}_i)$$

and receive rewards (and costs) exclusively determined by the products that $\mathrm{IWN}_i$ delivers to $\mathrm{M}_i$. However this would lead the agents to compete rather than cooperate and would not be coherent with a global approach, in which decisions at the three locations must aim at optimizing the whole system performance.

Note that the decision process is a discrete-state, continuous-time decision process. In fact, even though decision epochs are discrete as they occur when a new demand arises, the system may change state in-between decision epochs: for instance a transition may occur because of a delivery from a $\mathrm{PP}_i$ to an $\mathrm{EW}_i$. Therefore the decision process is a semi-markov decision process (SMDP). The learning process (SMART) applied on this SMDP has been simulated using SIMAN (Pegden *et al.*, 1990). Results are discussed in the next Section.

## 6. Results

Results concern (i) learning of coordination policies, which will be referred to as SMART policies, and (ii) the performance achieved through such coordination policies. The policies obtained by the SMART algorithm have been evaluated against two heuristics, namely local heuristics (LH) and balanced heuristics (BH). In the LH, every *j* th agent issues all the orders from the warehouse $\mathrm{IW}_j$ to the $\mathrm{EW}_i$ in the same country, namely the three countries run their manufacturing and distribution facilities independently from each other. In the BH, when the *j* th agent must make a decision, it issues the order to the $\mathrm{EW}_i$ currently characterized by the minimum number of unfulfilled orders. In case of two (or three) export warehouses with the same number of unfulfilled orders, the agents select first the local warehouse ($i = j$), then the one where the average currency exchange rate is lower.

As shown in Table 3, three cases with different demand patterns, all with same demand rates, have been analyzed.

| | $\mu$ — $\sigma$ | | |
|---|---|---|---|
| | $i = 1$ | $i = 2$ | $i = 3$ |
| Pattern 1: $k = 1$ (CV = 1.00) | 5.831 — 5.831 | 2.915 — 2.915 | 1.458 — 1.458 |
| Pattern 2: $k = 5$ (CV = 0.45) | 5.831 — 2.608 | 2.915 — 1.304 | 1.458 — 0.652 |
| Pattern 3: $k = 10$ (CV = 0.32) | 5.831 — 1.844 | 2.915 — 0.922 | 1.458 — 0.461 |

*Table 3. The analyzed demand patterns: mean and standard deviation of time between demands.*

Note from Table 3 that, while the mean time between demands is constant, the variance values (and the coefficient of variations CVs) change with respect to the three values of the *k* parameter of the Erlang distribution. The mean and the variance of an Erlang distributed random variable are respectively

$$\mu = k\cdot\beta \text{ and } \sigma^2 = k\cdot\beta^2.$$

For the demand patterns that were examined, the learning process lasted for about 2000000 time units, which, for the assumed demand rates involves an average of 2400000 decision epochs (342857, 685714, 1371429 for agents 1 to 3 respectively). This required about five hours on a Pentium 200 PC. The trend of the average reward $\rho$ shows a similar shape for all the cases. Figure 4 refers to the Case: $k = 1$, for which the asymptotic value of $\rho$ is about equal to 184. However, slight differences exist in the learning strategies that were utilized, as the learning parameter and the exploratory rate have been fine tuned for each case. The asymptotic values of $\rho$ as well as the learned policies obviously differ among the cases.

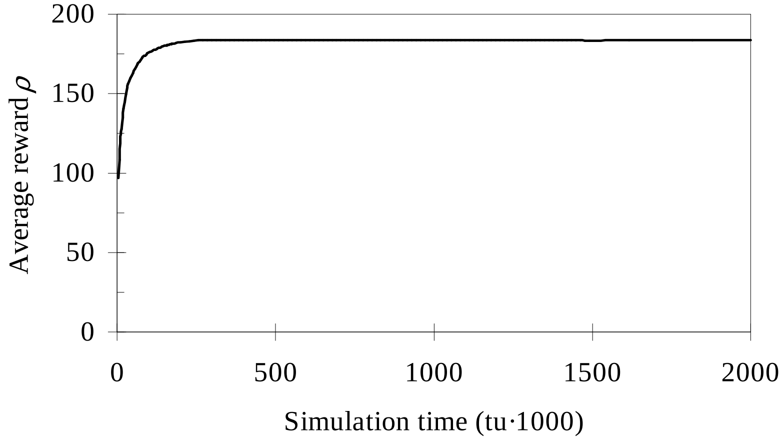


*Figure 4. Average reward trend during the simulated learning.*

Figure 5 compares the performances achieved by the policies (learned through SMART) and by the LH and BH policies, over a simulation run of 100000 time units.

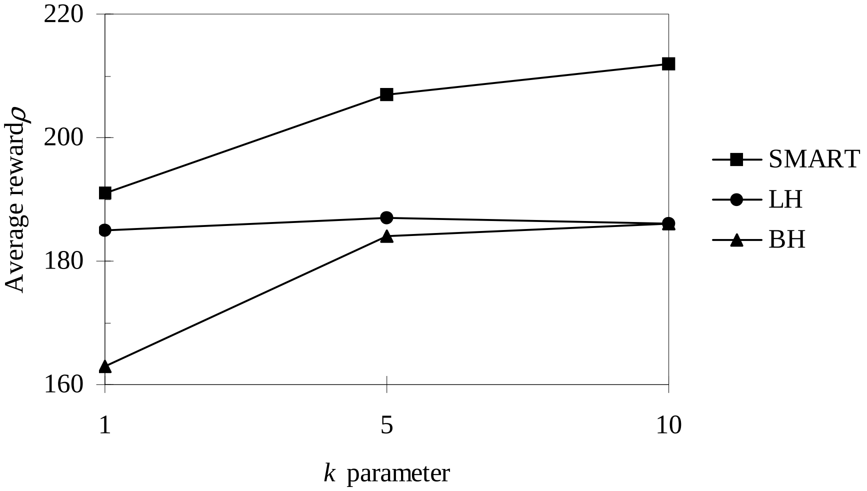


*Figure 5. Performance of the three policies for different demand uncertainty*

As expected, the LH policy is the least sensitive to demand uncertainty: in fact, as inter-country shipments are avoided under this policy, it involves lower lead times than the other policies. This in turn allows the system to quickly react to demands, thus keeping the average reward almost constant as uncertainty increases.

On the contrary, BH and SMART policies are very sensitive to uncertainty, because both policies allow inter-country shipments and thus incur higher lead times. In particular, when uncertainty is high ($k$ = 1), production shifting is not as worthwhile as synchronizing and smoothing operations: in fact, the BH policy performs much worse than LH, and the SMART policy results in slightly higher reward than the LH policy. When uncertainty is low ($k$ = 10), a higher emphasis on production shifting is needed. In such a case we find that the simple BH policy (which focuses only on coordination along the "area" dimension of the matrix in Figure 1) performs as good as the LH policy, whereas the SMART policy performs much better than both.

In summary, the results indicate that an integrated coordination policy should always be preferred. In fact, the three different policies learned by SMART under an integrated perspective are consistently better than the other policies. This is because they achieve the appropriate trade-off between production shifting and operations smoothing. In particular, it can be inferred that such a trade-off favors operations smoothing when uncertainty is high, which is evident from the performances of the LH and the SMART policies when k=1.

## 7. Conclusions

In this paper the management of distributed production systems has been studied. In particular, the analysis has addressed the coordination of multi-country production systems under the GSCM concept. A classification matrix has been proposed for characterizing the available approaches to the coordination problem in distributed systems. A methodology has been described to examine the coordination of distributed systems under the GSCM concept. The methodology, which is relatively new with regard to this particular field of application, is derived from three different areas of study, namely semi-Markov decision processes, reinforcement learning, and simulation. The reinforcement learning facilitates the solution of large semi-Markov decision problems through the use of agents that learn from experience. One specific feature of this reinforcement learning application is the use of multiple cooperating agents that share the same state space in pursuit of the same objective represented by a unique reward function for the whole system.

The results on a relatively simple case example show that the methodology is effective and achieves better performance than the two heuristics against which it was compared. Furthermore, the methodology derives different performance outcomes depending on market demand uncertainty. For high levels of demand uncertainty (modeled with high demand variance), any of the proposed coordination policies shows a decline in performance. However, the integrated policy derived from reinforcement learning (RL) approach outperforms any other policy. In fact, the RL based policy achieves the most appropriate trade-off between shifting production levels and smoothing of operations, and thus leads to maximization of the average reward. A key aspect of the methodology is the use of simulation, which makes it easy to apply the methodology to a wide range of problems. In fact, agents learn by simulating the real system, hence making it easy to model the system features. This aspect would be considerably more difficult if other approaches (e.g. dynamic programming techniques) were utilized. On the other hand, simulation by itself does not allow the determination of an *ex-ante* optimal solution of the problem.

Based on the foregoing, we believe that the proposed methodology represents a suitable tool for the coordination of distributed supply chain systems. Specifically, the methodology permits the modeling of variables that characterize international contexts, such as import tariffs and differences in real currency exchange rates. However, further research is needed to verify the effectiveness and veracity of the approach we proposed when dealing with larger and more complex GSCM problems.

## 8. References

1. Arntzen B.C., Brown G.G., Harrison T.P., Trafton L.L. (1995) 'Global supply chain at Digital Equipment Corporation', *Interfaces*, Vol. 25, No. 1, pp. 69-93.
2. Balakrishnan A., Kalakota, R., Whinston, A.B. and Ow, P.S. (1994) ‘Designing collaborative systems to support reactive problem-solving in manufacturing.’ In *Information and Collaboration Models of Integration*, edited by S.Y. Nof (The Netherlands: Kluwer) pp. 105-133.
3. Bartlett C.A., Ghoshal S. (1993) 'Managing across borders: new organizational responses', in Hedlund G. (ed.) *Organization of Transnational Corporations*, Routledge, London, pp. 326-342.
4. Bertsekas D., Tsitsiklis J. (1996) *Neuro-Dynamic programming*, Athena Scientific.
5. Cohen M.A., Fisher M., Jaikumar R. (1989) 'International manufacturing and distribution networks: a normative model framework', in Ferdows K. (ed.) *Managing International Manufacturing*, North Holland, New York, pp. 67-93.
6. Das T. K., Gosavi A., Mahadevan S., Marchalleck. N. (1999) 'Solving semi-markov decision problems using average reward reinforcement learning'. *Management Science*, Vol. 45, No. 4, pp. 560-574.
7. Doz Y. (1986) *Strategic Management in Multinational Companies*, Pergamon Press, Oxford.
8. Fawcett S.E., Birou L.M. (1992) 'Exploring the logistics interface between global and JIT sourcing', *International Journal of Physical Distribution & Logistics Management*, Vol. 22, No. 1, pp. 3-14.
9. Federgruen A. (1989) 'Methodologies for the evaluation and control of large scale production/distribution systems under uncertainty', in van Rijn C.F.H. (ed.) *Logistics: Where Ends Have to Meet*, Pergamon Press, New York, pp. 143-157.
10. Ferdows K. (1997) 'Making the most of foreign factories', *Harvard Business Review*, Vol. 75, No. 2, pp. 73-88.

11. Fisher M.L. (1997) 'What is the right supply chain for your product', *Harvard Business Review*, Vol. 75, No. 2, pp. 105-116.

12. Flaherty T.M. (1986) 'Coordinating international manufacturing and technology', in Porter M.E. (Ed.), *Competition in Global Industries*, Harvard Business School Press, Boston, pp. 83-109.

13. Gasser L. (1994). `Information and collaboration from a social/organizational perspective,' In *Information and Collaboration Models of Integration*, edited by S.Y. Nof (the Netherlands: Kluwer), pp. 237-261.

14. Huang C., Nof S.Y. (2000) `Formation of autonomous agent networks for manufacturing systems,' *International Journal of Production Research*, Vol. 38, No. 3, pp. 607-624.

15. Kogut B. (1990) 'International sequential advantages and network flexibility', in Bartlett C.A., Doz Y., Hedlund G. (eds.) *Managing the Global Firm*, Routledge, New York, pp. 47-68.

16. Kogut B., Kulatilaka N. (1994) 'Operating flexibility, global manufacturing, and the option value of a multinational network', *Management Science*, Vol. 40, No. 1, pp. 123-139.

17. LaLonde B.J., Masters J.M. (1990) 'Logistics: perspectives for the 1990s', *International Journal of Logistics Management*, Vol. 1, No. 1, pp. 1-6.

18. Lee H.L., Billington C. (1995) 'The evolution of supply-chain-management models and practice at Hewlett-Packard', *Interfaces*, Vol. 25, No. 5, pp. 42-63.

19. Levy D.J. (1997) 'Lean production in an international supply chain', *Sloan Management Review*, Vol. 38, No. 2, pp. 94-102.

20. Lootsma F.A. (1994) 'Alternative optimization strategies for large-scale production-allocation problems', *European Journal of Operational Research*, Vol. 75, pp. 13-40.

21. Oliff M.D., Arpan J.S., Dubois F.L. (1989) Global 'manufacturing rationalization: the design and management of international factory networks', in Ferdows K. (ed.) *Managing International Manufacturing*, North Holland, New York, pp. 41-65.

22. Pegden C.D., Shannon R.E., Sadowsky R.P. (1990) *Introduction to Simulation Using SIMAN*, McGraw-Hill, New York.

23. Pontrandolfo P. (1998) *Modelli per il coordinamento delle attività produttive nelle imprese multinazionali*, Ph.D. Thesis, Politecnico di Bari, Bari, Italy.

24. Pontrandolfo P., Okogbaa O.G. (1999) 'Global manufacturing: a review and a framework for planning in a global corporation', *International Journal of Production Research*, Vol. 37, No. 1, pp. 1-19.

25. Puterman M. (1994) *Markov Decision Processes: Discrete Stochastic Programming*, Wiley Interscience, New York.

26. Slats P.A., Bhola B., Evers J.J.M., Dijkhuizen G. (1995) 'Logistic chain modelling', *European Journal of Operational Research*, Vol. 87, No. 1, pp. 1-20.

27. Sutton R.L., Barto A.G. (1998) *Reinforcement Leaning - An Introduction*, MIT Press, Massachusetts.

28. Sutton R.S. (1988) 'Learning to predict by the methods of temporal differences', *Machine Learning*, Vol. 3, pp. 9-44.

29. Thomas D.J., Griffin P.M. (1996) 'Coordinated supply chain management', *European Journal of Operational Research*, Vol. 94, No. 1, pp. 1-15.

30. Vidal C.J., Goetschalckx M. (1997) 'Strategic production-distribution models: a critical review with emphasis on global supply chain models', *European Journal of Operational Research*, Vol. 98, No. 1, pp. 1-18.

## Appendix

### How the RL Algorithm (SMART) works

Consider a two state Markov chain with two actions allowed in each state. Let **TPM (a)** denote the one-step transition probability matrix associated with action *a*. Let **TRM (a)** denote the one-step transition reward associated with action *a*. Let the time spent in any transition be a uniformly distributed random variable from the distribution U (0.5,2.5).

**TPM (1)** = $\begin{bmatrix} 0.2 & 0.8 \\ 0.7 & 0.3 \end{bmatrix}$, **TRM (1)** = $\begin{bmatrix} 8 & -6 \\ 2 & 5 \end{bmatrix}$, **TPM (2)** = $\begin{bmatrix} 0.9 & 0.1 \\ 0.6 & 0.4 \end{bmatrix}$, and **TRM (2)** = $\begin{bmatrix} 6 & 1 \\ 3 & 7 \end{bmatrix}$.

Step 1: Initialize R($i$, $a$) = 0 for $i$=1, 2 and $a$=1, 2. Initialize C = 0 and T = 0. Hence $\rho$ =C /T=0. Initialize $m$ = 0. (Here $m$ will denote the number of jumps of the Markov chain.) Let the system simulation start in state 1. Since R(1, 1) = R(1, 2), both actions are equally good. Hence we choose any one. We select action 1 and simulate the action. Let the next state be 2. The learning rate will be defined by $\alpha_m$=0.01/$m$ and the exploration probability by $p_m$=0.1/$m$.

Step 2: The immediate reward must be –6. Also since $m$=1; we have $\alpha_{\mathrm{m}}$=0.01, $p_m$=0.1.and $\rho$ =0. Let the transition time be 1.4. Next, we update the R-value for the previous state action combination, i.e., state 1 and action 1.

R (1,1) $\leftarrow$ (1-$\alpha_m$) R (1, 1) + $\alpha_m$ [-6 – 0(1.4) + max{R (2, 1), R (2, 2)}] = -0.06.

C $\leftarrow$ C + (-6) = -6.

T $\leftarrow$ T + (1.4) = 1.4.

$\rho \leftarrow$ C/T = -4.286.

R (2,1) = 0 and R (2,2) = 0. Let the selected action be 2. Simulate action 2. Let the next state be 2.

Step 3: The immediate reward must be 7. Since $m$=2, we have $\alpha_m$=0.005, $p_m$=0.05 and $\rho$ = - 4.286. Let the transition time be 0.7. Next, we update an R-value of the previous state, i.e., state 2 and action 2.

R (2,2) $\leftarrow$ (1-$\alpha_m$) R (2, 2) + $\alpha_m$ [7 - (-4.287)(0.7) + max{R (2, 1), R (2, 2)}] = 0.5.

C $\leftarrow$ C + (7) = 1.

T $\leftarrow$ T + (0.7) = 2.1.

$\rho \leftarrow$ C / T = 1/(2.1) = 0.476

R (2,1) = 0 and R (2,2) = 0.5.

Hence 2 is the greedy action. An action other than the greedy action can be chosen with a probability of $p_m$. Let the selected action be 1. Simulate action 1. Let the next state be 1.

Step 4: The immediate reward must be 2. Since $m$=3, we have $\alpha_m$=0.00333, $p_m$=0.0333 and

$\rho$ = 0.476. Let the transition time be 2. Next, we update an R-value of the previous state, i.e., state 2 for the action selected.

R (2,1) ← (1-$\alpha_m$) R (2, 1) + $\alpha_m$ [2 – 0.476(2) + max{R (1, 1), R (1, 2)}] = 0.00329.

C ← C + (2) = 3.

T ← T + (2) = 4.1.

$\rho$ ← C/T = 0.731.

R (1,1) = -0.06 and R (1,2) = 0.

Hence 2 is the greedy action. A greedy action is selected with a probability of $p_m$. Let the selected action be 1. Simulate action 1. Let the next state be 2.

Step 5: The immediate reward must be -6. Since $m$=4, we have $\alpha_m$=0.0025, $p_m$=0.025 and $\rho$ = 0.731. Let the transition time be 1.9. Next, we update an R-value of the previous state, i.e., state 1 and action 1.

R (1,1) ← (1-$\alpha_m$) R (1, 1) + $\alpha_m$ [-6 – 0.731(1.9) + max{R (2, 1), R (2, 2)}] = -0.077.

C ← C + (-6) =3 .

T ← T + (1.9) = 6.

$\rho$ ← C/T = 0.5.

We continue visiting the states till the learning rate converges to 0 and no further updating of the R values is possible. The greedy actions at every state at that time constitute the policy learned by the algorithm.