Abstract

WEI, WENBIN. Quantifying Shared Information Value in a Supply Chain Using Decentralized Markov Decision Processes with Restricted Observations.

(Under the direction of Dr. Russell E. King and Dr. Thom J. Hodgson)

Information sharing in a two-stage and three-stage supply chain is studied. Assuming the customer demand distribution is known along the supply chain, the information to be shared is the inventory level of each supply chain member. In order to study the value of shared information, the supply chain is examined under different information sharing schemes. A Markov decision process (MDP) approach is used to model the supply chain, and the optimal policy given each scheme is determined. By comparing these schemes, the value of shared information can be quantified. Since the optimal policy maximizes the total profit within a supply chain, allocation of the profit among supply chain members, or transfer cost/price negotiation, is also discussed.

The information sharing schemes include full information sharing, partial information sharing and no information sharing. In the case of full information sharing, the supply chain problem is modeled as a single agent Markov decision process with complete observations (a traditional MDP) which can be solved based on the policy iteration method of Howard (1960). In the case of partial information sharing or no information sharing, the supply chain problem is modeled as a decentralized Markov decision process with restricted observations (DEC-ROMDP). Each agent may have complete observation of the process, or may have only restricted observation of the process. In order to solve the DEC-ROMDP, an *evolutionary coordination* algorithm is introduced, which proves to be effective if coupled with *policy perturbation* and *multiple start* strategies.

QUANTIFYING SHARED INFORMATION VALUE IN A SUPPLY CHAIN USING DECENTRALIZED MARKOV DECISION PROCESSES WITH RESTRICTED OBSERVATIONS

by

WENBIN WEI

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

INDUSTRIAL ENGINEERING

Raleigh, NC

2005

APPROVED BY:

Dr. R.E. King
Co-chair of Advisory Committee

Dr. T.J. Hodgson
Co-chair of Advisory Committee

Dr. H.L. W. Nuttle

Dr. J.M. Hughes-Oliver

Biography

Wenbin Wei was born in Wuhan, Hubei province, China, and grew up in Shiyan, an automobile city famous for its well-known Chinese trucks – "Dongfeng". Before he came to US in 2001 for his Ph.D. in the Industrial Engineering Department at North Carolina State University, he studied in Beijing for six and a half years in Beihang University (also known as Beijing University of Aeronautics and Astronautics), where he received his B.S. degree and M.S. degree from Mechanical Engineering and Automation School. During the summer of 2005, he worked as a technical student in SAS Institute Inc., Cary, NC.

Acknowledgements

I cordially thank my co-advisors, Dr. King and Dr. Hodgson. Their consistent support and patient guidance will be treasured for good. My special thank to my colleague Lauren Davis, whose dissertation research was closely related with mine. We spent plenty of time sharing programming codes and exchanging research insights. I also thank my colleague Kent Marshall, a robot expert, who has brought much joy to me.

I owe many thanks to my Mom, my Dad and my brother for their love and encouragement during so many years.

My thanks go to all my friends for their invaluable friendship and prayers. Especially those friends whom I made when I joined Wayne Ly's bible study group during 2001-2002. The time was exciting, fun and memorable. During 2003-2005 I joined International Bible Study (also called IBS) group near the NC State University campus. It has been enjoyable to share every Friday's evening at IBS getting away from work and worshiping God.

Table of Contents

List of Figures

List of Tables

Chapter 1 Overview

1.1 Introduction

If information in a supply chain is not shared among the individual chain elements (e.g., demand information), actual demand information (from downstream to upstream of the supply chain) may be distorted (this is also termed the bullwhip effect, Lee et al. 1997) and cause unnecessary cost. It has been reported that information sharing is beneficial to a supply chain, especially in reducing the bullwhip effect (Lee et al. 1997, 2000, Cachon and Fisher 2000) and supply chain cost (Gavirneni et al. 1999, Swaminathan et al. 1997, Tan 1999). However, it may not be beneficial to a supply chain if the cost of adopting the inter-organizational information system is too high (Swaminathan et al. 1997, Cohen 2000). In terms of information sharing, the concern is usually which production information to share and how to share it to maximize the mutual benefits in a supply chain (Huang et al. 2003).

The objective of this dissertation is to quantify the value of sharing inventory information in a make-to-stock environment and optimize the operational control for a two-stage and three-stage supply chain through appropriate information sharing. This dissertation is an extension of Davis' (2004) work on a two-stage supply chain with a single capacitated supplier and a single retailer. Davis finds the supplier's optimal policy by assuming the retailer uses a fixed policy, such as a base stock policy or (s, S) policy. Davis' work has some limitations. First, the retailer's policy is fixed. A more flexible policy could possibly achieve better system performance. Second, only the value of sharing retailer's inventory information is examined. This dissertation allows the retailer to use a flexible policy, and examines the value of sharing suppliers' inventory information. However, the problem becomes much more complicated since the suppliers and retailer need coordination when making their replenishment decisions in order to optimize the supply chain. Four different information sharing models are examined in a two-stage supply chain problem, while eight different models are examined in a three-stage supply chain.

Solving supply chain models for the optimal replenishment policy is a key to quantifying information value. Due to the difficulty of determining optimal policies for a

multi-echelon inventory system, researchers usually assert that a certain type of policy, like a base-stock policy, is optimal for one stage (Gavirneni et al. 1999, Gavirneni 2002, Simchi-Levi and Zhao 2002ab, Davis 2004) or both stages (Lee et al. 2000), and then find the specific policy for each stage. Those assumptions do not pursue system-wide optimality, since the assertion comes from the results of a single-stage inventory system, and the possibility of coordination between members is ignored. To overcome this drawback, we model a multi-stage supply chain as a Markov Decision Process (MDP).

In the context of a MDP, an agent with full observation (due to information sharing) actually faces a common MDP problem (also called a completely observable MDP, COMDP), while an agent with restricted observations (lack of information sharing) faces a MDP with restricted observations (called ROMDP). Davis (2004) solves a single agent ROMDP. As an extension, this dissertation provides a solution for multi-agent (decentralized) MDP or ROMDP problems (called DEC-ROMDP), where supply chain members need to be coordinated in order to maximize profit.

This dissertation proposes and analyzes an infinite horizon ROMDP with an average cost criterion, with an objective to maximize the average reward. A computationally efficient algorithm is developed to find optimal policies based on the policy iteration method of Howard (1960) for the infinite horizon undiscounted cost case. Formally, a ROMDP can be represented as a mixed integer nonlinear programming (MINLP) problem, for which it is difficult to find the global optimal solution. The basic heuristic proposed here includes two steps: value determination and policy improvement. It is proven that the policy improvement searches for an optimal solution by following a steepest ascent direction. We also propose perturbation methods, such as policy perturbation and Π perturbation (Π is the system steady state probability vector), to improve local optima towards a global optimal policy. Successive approximation is used to reduce computational effort. In addition, Ding's encapsulation evolution method (1985) can be used to further reduce computational effort for specially structured supply chain problems (Davis 2004).

A multi-agent model is viewed as a decentralized Markov decision process with restricted observation (DEC-ROMDP), which can be viewed as a special case of a decentralized POMDP (DEC-POMDP) (Bernstein et al., 2000) and a multi-agent team

decision problem (MTDP) (Pynadath and Tambe, 2002). An evolutionary coordination algorithm is used to make a joint policy evolve to a locally optimal solution, and then perturbation methods and a multiple restart strategy are used to improve the policy. By using the tools for solving DEC-ROMDP models, a wide range of supply chain problems with different information sharing schemes are solved.

Chapter 2 proposes the mathematical model for an infinite horizon ROMDP with an average cost criterion and introduces heuristic algorithms for solution. Chapter 3 gives the definition of DEC-ROMDP and proposes an evolutionary coordination algorithm to solve the multi-agent decision problems. Chapter 4 applies the evolutionary coordination algorithm to two-stage and three stage supply chain problems, and elaborates on information sharing and transfer cost negotiation within the supply chain. Chapter 5 outlines the future research to be performed.

1.2 References

Bernstein, D., S. Zilberstein, and N. Immerman, 2000. *The complexity of decentralized control of MDPs*. In Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence.

Cachon, G.P., and M. Fisher. 2000. Supply chain inventory management and the value of shared information. *Management Science*, 45, 843-856.

Cohen, S.L. 2000. Asymmetric information in vendor managed inventory systems. PhD Thesis, Stanford University.

Davis, L.B., 2004, State Clustering in Markov Decisions Processes with an Application in Information Sharing, unpublished *Ph.D. dissertation*, Industrial Engineering Department, N.C. State University.

Ding, F.Y., T. Hodgson and R. King, 1988. A methodology for computation reduction for specially structured large scale Markov decision problems. *European Journal of Operational Research*. *Volume* 34:105-112.

Gavirneni, S., R. Kapuscin, and S. Tayur. 1999. Value of information in capacitated supply chains. *Management Science*, 46, no.1: 16-24.

Gavirneni, S. 2002. Information flows in capacitated supply chains with fixed ordering cost. *Management Science*, 48, 644-651.

Howard, R. 1960. *Dynamic Programming and Markov Processes*. MIT Press, Cambridge, MA

Huang, G.Q., J.S.K. Lau, and K.L. Mak. 2003. The impacts of sharing production information on supply chain dynamics: a review of the literature. *International Journal of Production Research*. 41, no.7: 1483-1517.

Lee, H., Padmanabhan, V. and Whang, S. 1997. Information distortion in a supply chain: the bullwhip effect. Management Science, 43, 546-558

Lee, H.L., K.C. So, and C.S. Tang. 2000. The value of information sharing in a two-level supply chain. *Management Science*, 46, 626-643.

Pynadath, D., and M. Tambe, 2002. *The communicative multiagent team decision problem*: *Analyzing teamwork theories and models*. Journal of Artificial Intelligence Research, 16, 389-423.

Simchi-Levi, D., and Y. Zhao. 2002a. The value of information sharing in a two-stage supply chain with production capacity constraints. Working paper, Northwestern University, Evanston, IL.

_____. 2002b. The value of information sharing in a two-stage supply chain with production capacity constraints: the infinite horizon case. *Manufacturing and Service Operations Management*, 4, no.1: 21-24.

Swaminathan, J. M., N.M. Sadeh. and S.F. Smith. 1997. Effect of sharing supplier capacity information. Haas School of Business, University of California, Berkeley.

Chapter 2 Markov Decision Processes with Restricted Observations

2.1 Introduction

This chapter presents a computationally efficient procedure to determine control policies[1] for an infinite horizon, undiscounted Markov decision process (MDP) with restricted observations (ROMDP). In the MDP framework, it is usually assumed that an agent interacts synchronously with a world (Kaelbling, Littman, and Cassandra 1998). A Markov decision process can be defined as a tuple $< S, A, T, R >$. S is a finite set of $|S|$ world states; A is a set of $|A|$ actions; T: $S \times A \times S \rightarrow [0, 1]$ is the state-transition model, where $T(s, a, s')$ represents the probability of transition from state s to s', given that the agent takes action a; R: $S \times A \rightarrow R$ is the reward model, where $R(s, a)$ represents the expected reward for taking action a in state s (assumed to be bounded in this chapter). In a common MDP, the world state is assumed as completely observable to the agent, so this process is also called COMDP (completely observable Markov decision process) in this dissertation. Since this process is considered observable and the state of the system is observable to the agent, the stationary policy is a function of the state space.

If the world state is not completely observable to the agent, this process is a partially observable Markov decision process (POMDP), which can be defined as a tuple $<S, A, T, R, Z, O>$, where S, A, T, and R are the same as those in a COMDP. Z is a finite set of $|Z|$ observations; O: $S \times A \times Z \rightarrow [0, 1]$ is the observation probability distribution model, where $O(z, a, s')$ represents the probability that the agent observes z given that it took action a and then the world state reached s'. As the agent cannot observe the state directly, a POMDP policy, different from a COMDP policy, is not a function of the state space, but the function of belief states, i.e., the steady state probability distribution. Since the belief state is continuous, it is not realistic to find a policy based upon every possible belief state. However, an optimal policy can be based only upon finite partitions of belief

[1] This dissertation attempts to find the optimal stationary deterministic policy for an infinite horizon undiscounted ROMDP problem. A ROMDP policy space is a subset of a common MDP policy space. A common MDP policy can be categorized as deterministic or randomized, Markovian or history-dependent. A stationary policy is generally for an infinite horizon MDP problem. Refer to Puterman (1994) for details of these policy types.

state space (Smallwood and Sondik 1973)[2]. Several algorithms have been developed to efficiently determine the partitions (Sondik 1971, Cheng 1988, Littman 1994, and Zhang and Liu 1996).

A Markov decision process with restricted observation is a special POMDP, and it can be represented by a tuple $< S, A, T, R, Z, G >$, where S, A, T, R, and Z are the same as those in a POMDP. G: $S \rightarrow Z$ represents the mapping function from a state to a single observation for the agent. If a state s outputs an observation z, it can be denoted as $G(s) = z$. A ROMDP policy is represented as a function of the observation space. In this policy, if an action a is applied given an observation z, this action a would apply to any possible state s satisfying $G(s) = z$, that is, the action a must be implementable/admissible to all these states. Hence, a ROMDP policy is also called an "implementable policy" (Serin and Kulkarni 1995) or "admissible policy" (Smith 1971).

Although an ROMDP is a special POMDP, it is still intractable to solve. Serin and Kulkarni (1995) develop an algorithm that finds local optimal stationary randomized policies for the infinite horizon discounted reward case, with the objective to optimize the total discounted reward. Serin and Avsar (1997) introduce a similar algorithm for the finite horizon discounted reward case, and prove a deterministic optimal policy exists in this case. Smith (1971), Hordijk and Loeve (1994), and Hastings and Sadjadi (1979) present algorithms that determine deterministic policies for infinite horizon undiscounted reward problems, with the objective to optimize the average reward. The algorithm developed by Hastings and Sadjadi (1979) is enumerative based and thus intractable for large problems. The algorithm developed by Smith (1971) is a policy iteration type of algorithm containing enumerative component when a better policy cannot be determined. None of the above algorithms have addressed to an infinite horizon large scale ROMDP problem. This chapter introduces a computationally efficient algorithm that also finds optimal deterministic policies, based on the policy iteration method developed by Howard (1960) for the infinite horizon undiscounted cost case. This chapter demonstrates empirically that the algorithm finds the optimal deterministic policy for over 99% of the general ROMDP problem instances generated. In the instances where the optimal policy cannot be determined, the average error in the objective function is

[2] This applies to a finite horizon POMDP problem. It may not be the case for an infinite horizon problem.

less than 1%. This algorithm achieves better performance for supply chain ROMDP problems.

2.2 Mathematical Model for a ROMDP

2.2.1 ROMDP Notation

The process being analyzed is a Markov Decision Process with state space S and action space A. The state of the system cannot be observed, however some output of the system is observable. Based on those outputs, one can infer the state or possible states of the system. This chapter finds an optimal control policy defined over the observation process that maximizes the long term average reward. The optimal policy has the property that each state within a given observation set takes the same action. A summary of the problem notation is presented below.

S: The set of possible states $\{1\ldots N\}$.

A: The set of available actions $\{1\ldots M\}$.

X_n: A random variable that defines the state at time n.

Y_n: A random variable that defines the action by the agent at time n.

p_{ij}^a: The one step transition probability from state i to j given an action a.

$$p_{ij}^a = P\{X_{n+1} = j \mid X_n = i, Y_n = a\}, \forall i, j \in S, \forall a \in A,$$

c_{ia}: The expected immediate reward associated with transitioning to state i given action a.

Z: The set of observable outputs $\{1...K\}$.

$G(i)$: A function mapping a state i to a single observable output in the set Z.

S_k: A partition of the state space S satisfying $\{i: G(i) = k\}$. Without loss of generality, it is assumed any state space partition has the same number (say L) of states. Obviously, $K*L = N$, and $S_k = \{(k\text{-}1)*L+1, (k\text{-}1)*L+2\ldots k*L\}$, $k \in Z$.

$A(k)$: The set of admissible actions for the observation set S_k. Obviously, $A(k) \subseteq A$. Without loss of generality, it is assumed $A(k) = A$ for a generic ROMDP.

Z_n: A random variable that defines the observation by the agent at time n.

2.2.2 LP Model for a MDP Problem

Before presenting the mathematical model for a ROMDP with undiscounted reward and infinite horizon, this section starts from the linear programming model (LP) for an underlying common MDP problem (Wolf and Dantzig, 1962).

$$\max \sum_{a=1}^{M}\sum_{i=1}^{N} c_{ia} x_{ia}$$

subject to

$$\sum_{a=1}^{M} x_{ia} = \sum_{a=1}^{M}\sum_{j=1}^{N} x_{ja}\, p_{ji}^{a},\ \forall i \in S$$

$$\sum_{a=1}^{M}\sum_{i=1}^{N} x_{ia} = 1$$

$$x_{ia} \geq 0\ \forall i \in S, a \in A$$

This primal problem aims to maximize the average reward. Its decision variable x_{ia} can be interpreted as the steady state probability that state *i* will be visited at a typical transition and action *a* will be applied. The constraints can be satisfied with some feasible steady state probabilities associated with a certain randomized stationary policy, that is, a policy that chooses at state *i* the action *a* with probability x_{ia}.

LP theory implies that the optimal solution is always obtained with deterministic stationary policies. Indeed, if α* is an optimal (deterministic) stationary policy that is uni-chain ($\alpha^*(i)$ denotes the action for the state *i*), and x_i* is the corresponding steady state probability of state *i*, then $x_{ia}^{*} = \begin{cases} x_i^{*} & \text{if } a = \alpha^*(i) \\ 0 & \text{otherwise.} \end{cases}$ is an optimal solution of the primal problem.

It is also insightful to investigate the following dual formulation for the problem.

$$\min g$$

subject to

$$g + v_i - \sum_{j=1}^{N} p_{ij}^{a} v_j \geq c_{ia}\ \forall i \in S, a \in A$$

$$v_j \text{ free}, \forall j \in S$$

Howard's (1960) dynamic programming algorithm solves a COMDP problem from a dual perspective. This chapter solves the associated ROMDP from a primal perspective.

2.2.3 NLP Model for a ROMDP Problem

By adding observability constraints to the above primal MDP problem, the nonlinear programming (NLP) model for a corresponding ROMDP problem is obtained.

Before formulating the NLP model, the following definitions are introduced.

$\alpha = (\alpha_{11}, \alpha_{12}, ..., \alpha_{1M} \mid \alpha_{21}, ..., \alpha_{2M} \mid ... \mid \alpha_{K1}, ..., \alpha_{KM})$ is a ROMDP policy, in which αka denotes the probability that action a is applied given observation k. Here

$$\alpha_{ka} = \frac{x_{ia}}{\sum_{a=1}^{M} x_{ia}} = \frac{x_{ia}}{x_i}, \forall i \in S_k, a \in A.$$

$$x_i = \sum_{a=1}^{M} x_{ia}, \forall i \in S.$$

The NLP model for the ROMDP is given as

$$\max \sum_{i=1}^{N} \sum_{a=1}^{M} c_{ia} x_i \alpha_{G(i)a}$$

subject to

$$x_i = \sum_{j=1}^{N} \sum_{a=1}^{M} x_j \alpha_{G(j)a} p_{ji}^{a}, \forall i \in S$$

$$\sum_{i=1}^{N} x_i = 1$$

$$\sum_{a=1}^{M} \alpha_{ka} = 1, \forall k \in Z$$

$$x_i \geq 0 \ \forall i \in S$$

Let $P(\alpha)$ be the matrix defined with entries $p_{ij}(\alpha)$, where $p_{ij}(\alpha) = \sum_{a=1}^{M} \alpha_{G(i)a} p_{ij}^{a}$, and $c_i(\alpha) = \sum_{a=1}^{M} \alpha_{ka} c_{ia}$.

The NLP can be written in matrix notation as

$$\max \Phi(\alpha) = \bar{x}c(\alpha)$$
subject to
$$\bar{x}[I - P(\alpha)] = 0$$
$$\sum_{i=1}^{N} x_i = 1$$
$$\sum_{a=1}^{M} \alpha_{ka} = 1, \forall k \in Z$$
$$x_i \geq 0, \forall i \in S$$

The matrix [*I*-*P*(α)] is not invertible since it contains a redundant constraint. To reduce this redundant constraint, replace the Nth column of this matrix with all ones and define an invertible matrix $Q(\alpha)$.

$$Q(\alpha) = \begin{bmatrix} 1 - p_{11}(\alpha) & - p_{12}(\alpha) & \ldots & 1 \\ - p_{21}(\alpha) & 1 - p_{22}(\alpha) & \ldots & 1 \\ \ldots & \ldots & \ldots & 1 \\ - p_{N1}(\alpha) & - p_{N2}(\alpha) & \ldots & 1 \end{bmatrix}$$

Furthermore, define an n-element vector $\bar{b}$ =(0,0…1).

Then transform the NLP into

$$\max \Phi(\alpha) = \bar{x}c(\alpha)$$
subject to
$$\bar{x}Q(\alpha) = \bar{b}$$
$$\sum_{a=1}^{M} \alpha_{ka} = 1, \forall k \in Z$$
$$x_i \geq 0 \ \forall i \in S$$

By removing the variable *x*, it becomes

$$\max \Phi(\alpha) = \bar{b}Q(\alpha)^{-1}c(\alpha)$$
subject to
$$\sum_{a=1}^{M} \alpha_{ka} = 1, \forall k \in Z$$

Note the optimal solution to this NLP problem may be a randomized stationary policy α. That is, α_{ka}, a component of an optimal policy may be a number other than 0 and 1. It is intuitive that an optimal randomized policy will be better than an optimal deterministic policy. As only the optimal deterministic policy is in concern, the NLP is

transformed into a mixed integer nonlinear programming problem (MINLP) by adding the integer constraint.

$$\max \Phi(\alpha) = \bar{b}\, Q(\alpha)^{-1} c(\alpha)$$
subject to
$$\sum_{a=1}^{M} \alpha_{ka} = 1, \forall k \in Z$$
$$\alpha_{ka} \in \{0,1\}, \forall k \in Z, \forall a \in A$$

2.3 Methodology for Solving a ROMDP

Definition 1: $\beta = (\beta_{11}, \beta_{12}, ..., \beta_{1M} \mid \beta_{21}, ..., \beta_{2M} \mid ... \mid \beta_{K1}, ..., \beta_{KM})$ is a ***feasible direction*** at a feasible policy α if and only if $\alpha+\theta\beta$ is also a feasible policy for some θ>0.

Clearly, $\sum_{a=1}^{M} \beta_{ka} = 0, \forall k \in Z$ and $\beta_{ka} \geq 0$ for $\alpha_{ka} = 0$ and $\beta_{ka} \leq 0$ for $\alpha_{ka} = 1$. Without loss of generality, the normalization restriction $\sum_{k=1}^{K}\sum_{a=1}^{M} |\beta_{ka}| = 1$ is assumed on the feasible direction (Serin and Kulkarni 1995).

Definition 2: A feasible direction β is an ***ascent direction*** at a feasible policy α if $\Phi(\alpha + \theta\beta) > \Phi(\alpha)$, for all θ∈(0,δ) for some δ>0.

Lemma 1: If a feasible direction β satisfies that $\nabla\Phi(\alpha)^T \beta$ is positive, then β is an ascent direction at policy α. Here, $\nabla\Phi(\alpha)$ is the gradient of the objective function at α.

Proof: It is Obvious according to the definition of the gradient. (Q.E.D)

Theorem 1: Let $\bar{v} = [v_1 ... v_n]$ be the solution to $Q(\alpha)\boldsymbol{v}$= c(α), $\bar{x} = [x_1 ... x_n]$ be the solution to $\bar{x}Q(\alpha) = \bar{b}$, and P^{ka} is the matrix with $p_{ij}^{ka} = \begin{cases} p_{ij}^{a} & \text{if } G(i) = k, \text{and } j \neq N \\ 0 & \text{Otherwise} \end{cases}$. If

$a_k^* = \arg\max_{a \in A}(\sum_{i \in S_k} x_i (c_{ia} + \sum_{j=1}^{N} p_{ij}^{ka} v_j))$, then β satisfying $\beta_{ka} = \begin{cases} 1/2, \text{if } \alpha_{ka} = 0, \text{and } a = a_k^* \\ -1/2, \text{if } \alpha_{ka} = 1, \text{and } a \neq a_k^* \\ 0, \text{otherwise} \end{cases}$

is a ***steepest ascent direction*** at the current policy α, which maximizes the directional derivative $\nabla\Phi(\alpha)^T \beta$.

Proof:

First derive the gradient of the objective function at α, i.e. $\nabla\Phi(\alpha)$. To compute the gradient, it is assumed that the objective function is continuous and differential at every point of feasible region, i.e., α can be randomized.

$$\frac{\partial\Phi(\alpha)}{\partial\alpha_{ka}} = \sum_i \frac{\partial c_i(\alpha)}{\partial\alpha_{ka}} x_i + \sum_i \frac{\partial x_i}{\partial\alpha_{ka}} c_i(\alpha)$$

$$\frac{\partial c_i(\alpha)}{\partial\alpha_{ka}} = \frac{\partial}{\partial\alpha_{ka}}\left(\sum_a c_{ia}\alpha_{ka}\right) = \begin{cases} c_{ia} & \text{if } G(i)=k \\ 0 & \text{otherwise} \end{cases}$$

So $\frac{\partial\Phi(\alpha)}{\partial\alpha_{ka}} = \sum_{i\in S_k} c_{ia} x_i + \sum_i \frac{\partial x_i}{\partial\alpha_{ka}} c_i(\alpha)$

Let $\frac{\partial\bar{x}}{\partial\alpha_{ka}} = \left[\frac{\partial x_1}{\partial\alpha_{ka}}, \frac{\partial x_2}{\partial\alpha_{ka}} \ldots \frac{\partial x_N}{\partial\alpha_{ka}}\right]$

$$\frac{\partial\bar{x}}{\partial\alpha_{ka}} Q(\alpha) + \bar{x}\frac{\partial Q(\alpha)}{\partial\alpha_{ka}} = 0,$$

Define $P^{ka} = -\frac{\partial Q(\alpha)}{\partial(\alpha_{ka})}$. Clearly, P^{ka} is the matrix with

$$p_{ij}^{ka} = \begin{cases} p_{ij}^{a} & \text{if } G(i)=k, \text{and } j \neq N \\ 0 & \text{Otherwise} \end{cases}.$$

Then $\frac{\partial\bar{x}}{\partial\alpha_{ka}} Q(\alpha) = \bar{x}P^{ka}$. Since $Q(\alpha)^{-1}$ exists, $\frac{\partial\bar{x}}{\partial\alpha_{ka}} = \bar{x}P^{ka}Q(\alpha)^{-1}$.

Therefore, $\frac{\partial\Phi(\alpha)}{\partial\alpha_{ka}} = \sum_{i\in S_k} c_{ia} x_i + \bar{x}P^{ka}Q(\alpha)^{-1}c(\alpha)$

Let $\bar{v} = [v_1 \ldots v_n]$ be the solution to $Q(\alpha)\boldsymbol{v} = c(\alpha)$, Then

$$\frac{\partial\Phi(\alpha)}{\partial\alpha_{ka}} = \sum_{i\in S_k} x_i\left(c_{ia} + \sum_j p_{ij}^{ka}(\alpha)v_j\right).$$

In order to find the steepest ascent direction at the current policy α, $\nabla\Phi(\alpha)^T\beta$ should be maximized. Note the current policy α must be deterministic, for instance, if α uses action b for a state set S_k, there must be $\alpha_{ka} = \begin{cases} 1, & \text{if } a=b \\ 0, & \text{otherwise} \end{cases}$. It is obvious for a state set S_k, $(\beta_{k1},\ldots,\beta_{kM})$ must have a single negative component, say β_{kb}. Note

$\beta_{ka} \geq 0$ for $\alpha_{ka} = 0$, $\beta_{ka} \leq 0$ for $\alpha_{ka} = 1$, and $\sum_{a=1}^{M} \beta_{ka} = 0, \forall k \in Z$. In order to maximize $\nabla\Phi(\alpha)^T \beta$, it is obvious to choose a component β_{ka*} among $(\beta_{k1},..., \beta_{kM})$, where $a^* = \arg\max_{a \in A}(\frac{\partial\Phi(\alpha)}{\partial\alpha_{ka}})$, and set β_{ka*} to be $-\beta_{kb}$, other components zero.

Considering the normalization restriction $\sum_{k=1}^{K}\sum_{a=1}^{M} |\beta_{ka}| = 1$, so $\beta_{ka*} = 1/2$, and $\beta_{kb} = -1/2$. If $a^* = b$, every component of $(\beta_{k1},..., \beta_{kM})$ is zero. (Q.E.D)

If a steepest ascent direction β at a current policy α is zero, the policy α is considered as a local optimal policy. Otherwise, if β is not zero, there exists a step size $\theta>0$ such that $\alpha+\theta\beta$ (it might be a randomized policy) is better than α, i.e. $\Phi(\alpha+\theta\beta)>\Phi(\alpha)$. Since α is deterministic and the steepest ascent direction β is defined as Theorem 1, it is necessary to make $\theta = 2$ such that $\alpha' = \alpha+\theta\beta$ is also deterministic. Note a step size $\theta = 2$ may not satisfy $\Phi(\alpha+\theta\beta)>\Phi(\alpha)$. However, it is sufficient to keep this step size and ensure policy move between deterministic policies. This movement has a drawback that may not keep $\Phi(\alpha+\theta\beta)>\Phi(\alpha)$, but if that is a case, the current policy α is also considered as a local optimal policy. Based on this guidance, a heuristic algorithm is presented below.

Definition 3: If starting from a current policy α, β is the steepest ascent direction found according to Theorem 1, then the operation of getting a new policy $\alpha' = \alpha+2\beta$ is called a ***policy improvement***.

Definition 4: A policy α is a ***local optimal solution***, if after a policy improvement the policy change from α to α', and $\Phi(\alpha') \leq \Phi(\alpha)$.

Lemma 2: Let $\bar{v} = [v_1 ... v_N]$ be the solution to $Q(\alpha)\boldsymbol{v} = c(\alpha)$, then $g = v_N$ is the gain associated with the policy α.

Proof: For a policy α, the gain is $g = \bar{x}c(\alpha)$, where x is the steady state probability. Since $\bar{x} = \bar{b}Q(\alpha)^{-1}$, the gain is $g = \bar{b}Q(\alpha)^{-1}c(\alpha) = \bar{b}v$. Hence $g = v_N$ as $b = [0,0, \ldots,1]'$. (Q.E.D)

As deterministic policies are of interest, it is sufficient to only carry the information needed in terms of the action taken for a given observation set S_k, and this dispels the necessity to construct a *K*M*-element decision vector α to represent an implementable policy. Therefore, a *K*M*-element original decision vector α can be represented by *K*-element policy vector $\bar{\delta} = [\delta_1 ... \delta_K]$, where $\delta_k = a$ if $\alpha_{ka} = 1$.

Then $Q(\alpha)$ has entries $p_{ij}(\alpha)$ where

$$p_{ij}(\alpha) = \sum_a \alpha_{G(i),a} p_{ij}^a = p_{ij}^{\delta_{G(i)}}$$

Similarly, the vector $c(\alpha)$ has entries $c_i(\alpha)$ where

$$c_i(\alpha) = \sum_a c_{ia} \alpha_{G(i),a} = c_{i,\delta_{G(i)}}$$

These substitutions will be denoted as $Q(\alpha/\delta)$ and $c(\alpha/\delta)$. To simply notation, a policy is always considered as *K*-element vector notation in this chapter, but readers should keep in mind that the policy can have two representations. Use α_k to represent the action used for observation set *k*.

Heuristic Algorithm: The algorithm for finding an implementable policy is as below.

Step 0. Initialization

Generate an initial admissible policy α.

Set $g^* = -\infty$.

Step 1. Value Determination

Determine relative values *v*, steady state probabilities *x*, and the gain g^δ.

$\bar{x} = \bar{b}\, Q(\alpha)^{-1}$, $\boldsymbol{v} = Q(\alpha)^{-1} c(\alpha)$,

(*a*). If $g^\alpha > g^*$, set $g^* = g$ and proceed to Step 2.

(*b*) If $g^\alpha \le g^*$, the current solution is a local maximum, and stop.

Step 2. Policy Improvement

For all $k \in Z$ find an action $\alpha_k = \arg\max_{a \in A} \sum_{i \in S_k} x_i (c_{ia} + \sum_{j=1}^{N} p_{ij}^{ka} v_j)$, and

go to step 1.

Lemma 3: The algorithm defined above will terminate at a local optimal solution after a finite number of iterations.

Proof: By assuming the reward for the Markov decision process is bounded, the gain associated with any policy would be bounded. Since the policy improvement step will not increase the gain indefinitely, there must be a finite number of iterations such that the algorithm terminates. (Q.E.D)

2.4 Perturbation

The above heuristic, called *Normal Convergence*, does not guarantee to obtain the global optimum unless only a single local optimum exists. Therefore, the normal convergence is augmented by a local improvement procedure, called *perturbation*, to increase the probability of finding the global optimum.

In order to improve the heuristic, this chapter also provides two perturbation methods. One is called *Policy Perturbation*, and the other is called Π *Perturbation*.

2.4.1 Policy Perturbation

Policy perturbation is carried out based on the policy from Normal Convergence. The basic idea is to perturb the best policy obtained from normal convergence, and form a neighboring policy. By starting from this new policy, repeat value determination and policy improvement cycle. Once a better policy is found, continue perturbing this policy until no better policy can be found.

Obviously, how a policy is perturbed and how many perturbations are performed impacts the effect of policy perturbations. Two approaches for policy perturbation are developed. The first one (denoted as PP1) modifies only one element in the policy vector, and the number of the perturbation increases proportionally to the length of the policy vector. The second one (denoted as PP2) is an extension of the first one. After modifying one element in the policy vector, it tries to modify any adjacent two elements in the policy vector. Obviously, PP2 has more perturbations than PP1.

How to modify the element to obtain a new policy? During the policy improvement step, the test quantities for different alternatives are computed, and the best alternative is determined, which maximizes the test quantity, say,

$\sum_{i \in S_k} x_i (c_{ia} + \sum_{j=1}^{N} p_{ij}^{ka} v_j)$. Actually, the second best alternative can serve as the candidate for perturbation. The example below demonstrates the approach.

The original policy $\alpha = (\alpha_{11}, \alpha_{12}, ..., \alpha_{1M} \mid \alpha_{21}, ..., \alpha_{2M} \mid, ... \mid \alpha_{K1}, ..., \alpha_{KM})$ is not convenient to represent. According to the characteristic of deterministic policy α, this chapter simplifies the representation of this policy with K-element policy vector, of which each element corresponds to the selected action index for one of the K observations. For instance, for a ROMDP problem with $N = 16$, $M = 4$, $K = 4$, and $L = 4$, , if a best policy is (4, 3, 3, 2) after *Normal Convergence*, the original representation of policy is actually α=(0,0,0,1| 0,0,1,0| 0,0,1,0| 0,1,0,0). Assume the second best alternatives are (3, 4, 1, 3). Policy Perturbation I (denoted as PP1) results in the following policies after perturbation: (**3**,3,3,2), (4,**4**,3,2), (4,3,**1**,2), and (4,3,3,**3**). The Policy Perturbation II (denoted as PP2) results in the following policies after perturbation: (**3**,3,3,2), (4,**4**,3,2), (4,3,**1**,2), (4,3,3,**3**), (**3**,**4**,3,2), (4,**4**,**1**,2), (4,3,**1**,**3**), and (**3**,3,3,**3**). Notice that the first element and the last element in a vector are treated as adjacent.

2.4.2 Pi Perturbation

Pi Perturbation (Π *Perturbation*) is similar to the policy perturbation, except that it perturbs a steady state probability vector x instead of a policy vector. Although the policy is not modified, modification of vector x may lead to a better policy by repeating the value determination and policy improvement cycle. If a better policy is found, again perturb the vector x associated with this policy. The process is continued until no better policy can be found.

Unlike a policy vector, vector x has a continuous space. Different from policy perturbation, Π perturbation is performed by randomizing the x vector under the expectation that this modified vector x will eventually lead to a better policy during the value determination and policy improvement cycle.

Two types of Π perturbation are developed (set $\varepsilon = 1/N$).

(1) Π *Perturbation I* (denoted as PiP1)

i) $x_i = x_i + \varepsilon \ \forall i$

ii) $x_i = \dfrac{x_i}{\sum_{i=1}^{N} x_i} \quad \forall i$

(2) Π *Perturbation II* (denoted as PiP2)

$$\text{i) } x_i = \begin{cases} \max(0, x_i - \varepsilon) & \text{if } r \le 0.5, \forall i \\ x_i + \varepsilon & \text{if } r > 0.5, \forall i \end{cases},$$

where r is randomly generated value between 0 and 1.

$$\text{ii) } x_i = \frac{x_i}{\sum_{i=1}^{N} x_i} \quad \forall i$$

2.5 Experimentation

2.5.1 Generic Problem

To evaluate the effectiveness of an algorithm, different sizes of generic ROMDP problems are generated, and each problem has 1000 random instances. The heuristic solution of solving these instances was compared with the optimal solution through brute force enumeration. Let *K* represent the number of partitions, *L* the number of states in each partition, *N*=*K***L* the number of total states in the system, and *M* the size of action space.

Table 2.1 gives the percentage of problems solved optimally. By applying *Normal Convergence* (NC), *Policy Perturbation* I and II (PP1 and PP2), and Π Perturbation I and II (PiP1 and PiP2), 88.3%, 98.5%, 99.2%, 98.8%, and 99.1% of 1,000 problem instances are optimally solved, respectively. By combining PP1 and PiP2, 99.7% of 1,000 problem instances were solved optimally.

Table 2.2 gives the average error from the optimal solution for those problems that are not solved optimally. Table 2.3 gives the maximum error from the optimal solution for those problems that are not solved optimally. Table 2.4 gives the average execution time (seconds) for each problem. The results show that NC can optimally solve at least 85% of generic ROMDP problems. With policy perturbation or Π perturbation, more than 96% of these problems are solved optimally. Among those problems not solved optimally, the average errors are less than 2% and the maximum error are less than 10%; with perturbations, the errors are much smaller. As the size of the action space increases, the policy space increases exponentially and it is prohibitive to use brute force enumeration to obtain the optimal solution for larger problems. Our algorithm appears effective and fast to the generic ROMDP problem.

Table 2.1: Percentage of Problems Solved Optimally (Generic ROMDP)

% of Problem Solved Optimally Among 1000 Instances							
KXL, N, M	Policy space	NC	PP1	PP2	PiP1	PiP2	PP1&PiP2
3X3, 9, 3	3^3= 27	88.90%	97.40%	98.30%	96.20%	98.10%	98.90%
4X4, 16, 4	4^4=256	87.60%	98.30%	98.90%	97.70%	98.40%	99.50%
5X5, 25, 5	5^5= 3125	88.30%	98.50%	99.20%	98.80%	99.10%	99.70%
6X6, 36, 6	6^6= 46656	98.4%	99.6%	99.7%	98.7%	98.4%	99.6%

Table 2.2: Average error of Problems Not Solved Optimally (Generic ROMDP)

Average Error of Problems Not Solved Optimally						
KXL, N, M	NC	PP1	PP2	PiP1	PiP2	PP1&PiP2
3X3, 9, 3	1.54%	1.33%	1.18%	1.39%	0.79%	0.68%
4X4, 16, 4	0.69%	0.43%	0.45%	0.57%	0.40%	0.55%
5X5, 25, 5	0.34%	0.19%	0.25%	0.16%	0.21%	0.28%
6X6, 36, 6	0.09%	0.09%	0.09%	0.09%	0.09%	0.09%

Table 2.3: Max Error of Problems Not solved Optimally (Generic ROMDP)

Max Error of Problems Not Solved Optimally						
KXL, N, M	NC	PP1	PP2	PiP1	PiP2	PP1&PiP2
3X3, 9, 3	9.36%	5.20%	4.79%	7.23%	2.29%	1.81%
4X4, 16, 4	3.45%	1.44%	1.44%	1.86%	1.03%	1.44%
5X5, 25, 5	1.65%	0.64%	0.64%	0.77%	0.70%	0.64%
6X6, 36, 6	0.23%	0.16%	0.16%	0.23%	0.23%	0.16%

Table 2.4: Average Execution Time (seconds) (Generic ROMDP)

Average Execution Time (seconds[3]) for 1000 Problem Instances							
KXL, N, M	NC	PP1	PP2	PiP1	PiP2	PP1&PiP2	Enumeration
3X3, 9, 3	0.00026	0.0013	0.0025	0.0017	0.0024	0.0035	0.0014
4X4, 16, 4	0.0012	0.0063	0.011	0.0083	0.014	0.018	0.037
5X5, 25, 5	0.0030	0.024	0.043	0.037	0.059	0.073	1.5
6X6, 36, 6	0.01	0.08	0.16	0.08	0.23	0.30	74

2.5.2 Supply Chain Problem

The ROMDP algorithm is also applied to a two-stage supply chain ROMDP problem (maximization problem), in which the retailer uses a fixed order-up-to policy, and the supplier aims to optimize the system without knowing the retailer's inventory information.

[3] The experiments were performed on a computer with Intel Pentium 2.2GHz CPU.

The assumptions include: There is a customer demand distribution that retailer must satisfy. The supplier's production and the retailer's order shipment are synchronous, and their lead-time is a typical period. Each has a maximum inventory capacity. The supplier's production capacity is limited by its inventory capacity, since it cannot produce more than can be accommodated in his warehouse. The retailer applies order-up-to policy, and the order-up-to level is its inventory capacity. Note that the excess demand from a customer or the retailer is lost. The cost structure includes production/order setup cost (F_s and F_r), holding cost (H_s and H_r), variable production/purchase cost (W_s and W_r), and a stock out penalty cost (L_s and L_r). Here the subscription of "*s*" stands for the supplier and "*r*" for the retailer.

The typical parameters for the supply chain are as follows.

C_s : The inventory capacity for the supplier.

C_r : The inventory capacity for the retailer.

V: The selling price to the customer.

d : The demand from the customer, $d = 0,1 \ldots \mathrm{D}$, assuming $D = C_r$.

i_s : The inventory level of the supplier, $i_s = 0,1,2,...,C_s$. The supplier's observation on his own inventory is $z_s = i_s$.

i_r : The inventory level of the supplier, $i_r = 0,1,2,...,C_r$. The retailer's observation on her own inventory is $z_r = i_r$.

k_s : The production order quantity placed by the supplier. The possible order quantity depends on the supplier's inventory capacity and current inventory level, i.e., $k_s = 0,1,2,...,C_s - i_s$.

The objective is to find the optimal policy for the supplier, who only observes his own inventory, such that the supply chain total profit is maximize. Obviously, this is a typical ROMDP problem. The system state can be represented by the inventories of both the supplier and the retailer, i.e., $i = (i_s - 1) * C_s + i_r$, and the action can be represented by the order quantity of the supplier, i.e., k_s . Since the supplier has the capacity restriction, the policy space is not as large as the generic ROMDP with the same action space. Suppose the current state is *i* (the supplier and the retailer's inventories are

correspondingly i_s and i_r respectively), under an action k_s and a customer demand d, then the total profit of this supply chain would be:

$$P(i, k_s, d) = V * \min(d, i_r) - [H_s * i_s + H_r * i_r + W_s * k_s + \min(1, k_s) * F_s + \min(1, \min(k_r, i_s)) * F_r + L_r * (d - i_r)^+]$$

Since C_s and C_r determines the problem size, 1000 problem instances for different C_s and C_r are generated. It appears that the performance is better than generic problems (see Figure 2.1, Figure 2.2, and Figure 2.3). Note that $C_s + 1 = K$ and $C_r + 1 = L$. Without any perturbation, NC method has achieved more than 93% of problems solved optimally. With perturbation, almost solve all the problems are solved; even for those problems that are not solved optimally, the average errors are close to zero.

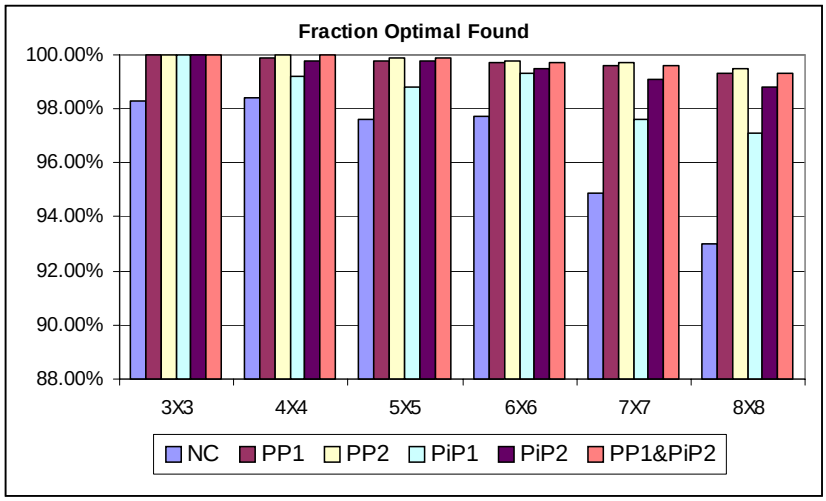

Figure 2.1: the percentage of problems solved optimally (Supply Chain ROMDP)

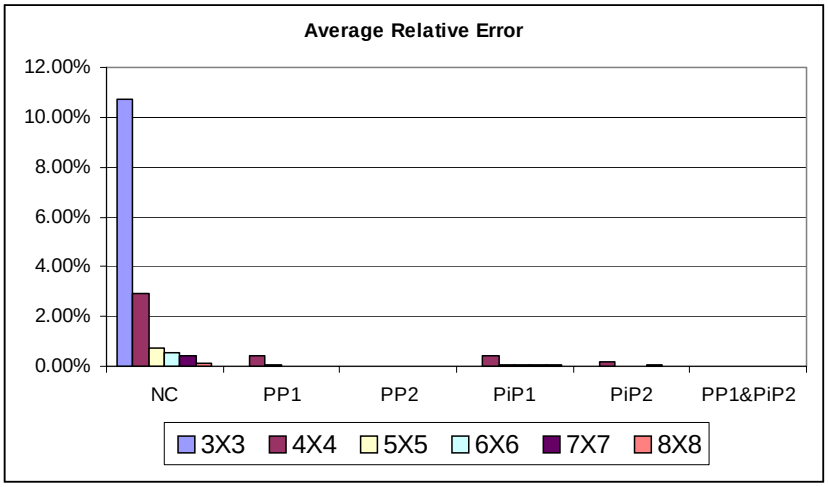

Figure 2.2: The Average error of Problems Unsolved Optimally (Supply Chain ROMDP)

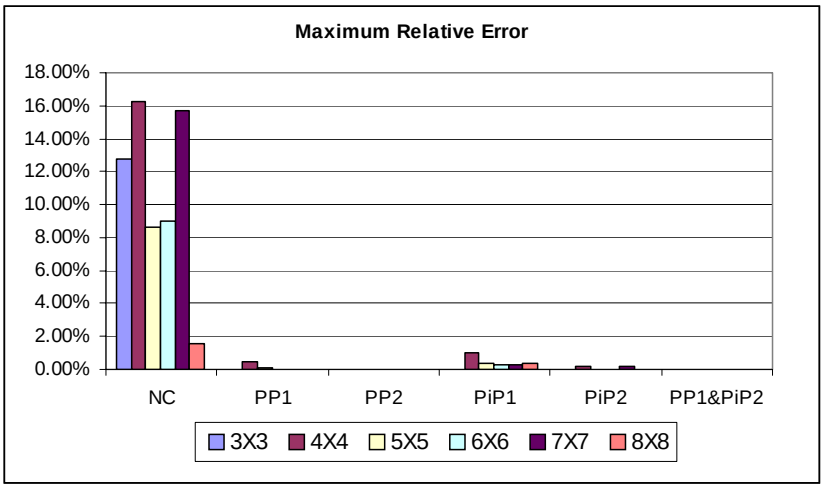

Figure 2.3: Max Error of Problems Unsolved Optimally (Supply Chain ROMDP)

Table 2.5: Average Execution Time (seconds) (Supply Chain ROMDP)

Average Execution Time (seconds) for 1000 Problem Instances							
KXL, N, M	NC	PP1	PP2	PiP1	PiP2	PP1&PiP2	Enumeration
3X3, 9, 3	8e-005	0.000251	0.000233	0.000128	7.9e-005	0.00022	4.7e-005
4X4, 16, 4	0.000252	0.000862	0.001359	0.000267	0.000204	0.000845	0.000623
5X5, 25, 5	0.00066	0.002282	0.002818	0.000661	0.000704	0.002072	0.007928
6X6, 36, 6	0.001128	0.004493	0.008472	0.001197	0.001108	0.005118	0.103613
7X7, 49, 7	0.002471	0.011151	0.019283	0.002475	0.002805	0.01094	1.43334
8X8, 64, 8	0.02791	0.14635	0.24389	0.31893	0.33917	0.43547	101.69725

2.6 Conclusion

Experimental results demonstrate that the heuristic approach to solving ROMDP problems is very effective and efficient. For practical supply chain problems, it has better performance. The heuristic approach can be used for solving large-scale ROMDP problems (Davis 2004).

2.7 References

Cheng, Hsien-Te. 1988. Algorithms for Partially observable Markov Decision Processes. PhD thesis, University of British Comubia, British Coumbia, Canada.

Davis, L.B., 2004, State Clustering in Markov Decisions Processes with an Application in Information Sharing, unpublished *Ph.D. dissertation*, Industrial Engineering Department, N.C. State University.

Hastings, N., and D. Sadjadi. 1979. Short communication: Markov programming with policy constraints. European Journal of Operational Research, 3:253-255.

Hordijk, A., and J. Loeve. 1994. Undiscounted Markov decision chains with partial information: an algorithm for computing a locally optimal periodic policy. Mathematical Methods of Operations Research, 40:163-181.

Howard, R. 1960. Dynamic Programming and Markov Processes. MIT Press, Cambridge, MA

Kaelbling, L.P., M.L. Littman, and A.R. Cassandra. 1995. Planning and acting in partially observable stochastic domains. Technical Report CS-96-08, Brown University, Providence, RI.

Littmann, M. 1994. The witness algorithm: Solving partially observable Markov decision processes. Technical report CS-94-40, Department of Computer Science, Brown University.

Serin, Y., and Z. Avsar. 1997. Markov decision processes with restricted observations: finite horizon case. Naval Research Logistics, 44: 439-456.

Serin, Y., and V.G. Kulkarni. 1995. “Implementable policies: discounted cost case” in W.J. Steward (Ed.), Computations with Markov Chains. Kluwer Academic Publishers, Dordrecht.

Smith, J.L. 1971. Markov decisions on a partitioned state space. IEEE transactions on systems, man and cybernetics SMC-1, no.1: 55-60

Sondik, E.J. 1971. The Optimal Control of Partially Observable Markov Processes. PhD thesis, Stanford University, Stanford, California.

Wolfe, P., and G.B. Dantzig. 1962. Linear programming in a Markov chain. Operations Research 10: 702-710.

Zhang, N.L., W. Liu. 1996. Planning in stochastic domains: Problem characteristics and approximations. Technical report HKUST-CS96-31.

Chapter 3 Decentralized Markov Decision Processes with Restricted Observations

3.1 Introduction

This chapter presents a computationally efficient algorithm to solve a distributed multi-agent decision process problem. It is assumed that a group of agents are fully cooperative, and that the objective is to derive optimal joint policies for the agents that maximize the joint reward over an infinite horizon.

Generally, a Markov Decision Process or MDP (Howard, 1960) can be used to model a single agent decision problem where the agent has full observability of the process. Within a multi-agent framework, the global state may not be observable by every agent. It is assumed that agents are only able to observe their local states which are the observable partitions of the global state space. Due to the partial observability, each agent faces a Restricted Observable Markov Decision Process or ROMDP (Chapter 2). It is instructional to note that a ROMDP is a special case of a partially observable Markov decision process or POMDP (Sondik, 1971). In a POMDP, for each global state there is a probability distribution associated with the resulting observation whereas in a ROMDP there is a single observation associated with each global state (although multiple global states may yield the same observation). Thus, the multi-agent problem can be viewed as a Decentralized ROMDP (DEC-ROMDP). A DEC-ROMDP can be viewed as a special case of a decentralized POMDP (DEC-POMDP) (Bernstein *et al*., 2000) and a multi-agent team decision problem (MTDP) (Pynadath and Tambe, 2002). Note that within a DEC-ROMDP framework, if every agent has full observability of the global state, the DEC-ROMDP degenerates into a Multi-agent MDP (MMDP) (Boutilier, 1999) or a Decentralized MDP (DEC-MDP) (Bernstein *et al*., 2000), where every agent is a MDP decision maker that collectively acts to achieve a common objective.

Solving a decentralized Markov decision problem is extremely difficult. The computational complexity of a DEC-POMDP with at least two agents or a DEC-MDP with at least three agents is complete for the complexity class nondeterministic exponential time (Bernstein et al., 2000). One approach to circumventing this complexity barrier is to exploit the structure of decentralized problems. For example, Becker et al.

(2002) present a coverage set algorithm to solve a general class of decentralized MDPs that exhibits transition independence without reward independence. Another approach is to simplify the nature of decentralized decision problems. For example, Chades et al. (2002) convert a DEC-POMDP into a MMDP (Boutilier, 1999) by approximating the reward function and transition function over observations instead of over states. However, the conversion from solving a DEC-POMDP to solving a MMDP can be quite complex and the solution to the MMDP is approximate to the DEC-POMDP since it ignores the nonstationary property of the transition and reward functions over observations.

Researchers have been exploiting algorithms within the framework of finite horizon DEC-POMDPs and DEC-MDPs (for example, Becker et al., 2002, Nair et al. 2003, Chades et al. 2003, Xuan et al., 2001). Chapter 2 presents an effective approach for solving single agent ROMDP problems. However, a DEC-ROMDP cannot be treated as separate ROMDPs because the transition and reward function generally depends on the joint policy, rather than a single agent policy. To the best of the authors' knowledge there is no efficient algorithm for DEC-ROMDPs in the literature.

This chapter presents an evolutionary coordination mechanism to evolve a joint policy to a locally optimal policy for infinite horizon DEC-ROMDPs. In the coordination mechanism, each agent iteratively updates their local policy while keeping the other agents' policies fixed. Each update attempts to increase the joint reward until no improvement can be made. Similar coordination mechanisms are studied by Nair et al. (2003) and Chades et al. (2002) for finite horizon DEC-POMDPs. For example, Nair et al. (2003) present a similar coordination mechanism called JESP (joint equilibrium-based search for policy) which uses either exhaustive search or dynamic programming to find the best policy for each agent.

Our experimentation indicates that the evolutionary coordination algorithm, coupled with a multiple start strategy and policy perturbation, effectively solves general DEC-ROMDPs. Additional experimentation shows that for specially structured supply chain problems modeled as infinite horizon DEC-ROMDPs, 100% of problems tested are solved optimally. Using successive approximation (White, 1960) to reduce computation

effort, this algorithm has been used to solve large-scale supply chain problems, and appears to be effective and efficient.

3.2 Model

3.2.1 Single agent MDP and ROMDP

A single agent Markov decision process can be defined as a tuple < *S, A, T, R* >. *S* is the finite set of global states; *A* is the set of actions; *T*: $S \times A \times S \rightarrow [0,1]$ is the state-transition model, where $p_{s,s'}^{a}$ represents the probability of ending at a state *s'* given that the process is in state *s* and the agent takes action *a*; *R*: $S \times A \rightarrow R$ is the reward model, where r_s^a [4] represents the expected reward when taking action *a* in state *s*. In a common MDP (referred henceforth as a completely observable Markov decision process or COMDP), the global state is assumed as completely observable to the agent.

If a global state is not completely observable to the agent, this process is a partially observable Markov decision process (POMDP), which can be defined as a tuple <*S, A, T, R, Z, O*>, where *S, A, T,* and *R* are the same as those in a COMDP. *Z* is the finite set of observations; *O*: $S \times A \times Z \rightarrow [0, 1]$ is an observation probability distribution model, where $o_{s',z}^{a}$ represents the probability that the agent observes *z* given that it took action *a* and then the global state changed to *s*'. If the observation probability distribution *O* is simplified as a mapping function such that $G(s)=z$, the POMDP degenerates into a ROMDP. Thus, a ROMDP can be represented by a tuple < *S, A, T, R, Z, G* >, where *S, A, T, R,* and *Z* are the same as those in a POMDP. *G*: $S \rightarrow Z$ represents the mapping function from a state to a single observation for the agent. Note that the mapping relationship ensures the partitioning of the state space by observations. Specially, if $G(s)=s$, the ROMDP degenerates into a COMDP.

This chapter finds the optimal stationary deterministic policy[5] to maximize average reward for infinite horizon decision problems. A COMDP policy can be represented as a function of the state space, and a ROMDP policy a function of the observation space. Under a RODMP policy, if an action *a* is applied given an

[4] Assumed bounded in this dissertation.
[5] A common MDP policy can be categorized as deterministic or randomized, Markovian or history-dependent. A stationary policy is generally sought for an infinite horizon MDP problem. Refer to Puterman (1994) for details of these policy types.

observation z, this action a applies to any possible state s satisfying $G(s)=z$, that is, the action a must be implementable/admissible to all these states. Hence, a ROMDP policy is also called an "implementable policy" (Serin and Kulkarni, 1995) or "admissible policy" (Smith, 1971). Obviously, a ROMDP policy space is a subset of a common COMDP policy space.

3.2.2 DEC-ROMDP (Multi-agent)

Definition 1. An n-agent DEC-ROMDP is defined as a tuple $<S, A, T, R, Z, G, \Lambda>$, where

- S is a finite set of global states;
- $A = A_1 \times ... \times A_n$ is a finite set of joint actions, with A_i indicating the individual action set by agent i;
- T: $S \times A \times S \rightarrow [0,1]$ is a state-transition model, where $p^a_{s,s'}$ represents the probability of ending at state s', given that the system state is s and each agent i follows their individual action a_i. The collection of individual actions, $(a_1,\ldots, a_n)$, form a joint action a;
- R: $S \times A \rightarrow R$ is a reward model, where r^a_s represents the immediate expected reward for taking joint action $a=(a_1,\ldots, a_n)$ when the system state is s;
- $Z = \{Z_1,...,Z_n\}$ is a finite set of observations, with Z_i indicating the individual observation set of agent i;
- $G = \{G_1,...,G_n\}$ is a set of mapping functions, with G_i: $S \rightarrow Z_i$ indicating an individual mapping function from a state to an observation by agent i; and
- $\Lambda = \{1 \ldots n\}$ is a set of n agents.

Definition 2: Given an n-agent DEC-ROMDP, a ***stationary individual policy*** for an agent i is defined as δ_i: $Z_i \rightarrow A_i$, or δ_i: $S \rightarrow A_i$ (due to the mapping function between a global state and an observation by the agent, i.e. G_i: $S \rightarrow Z_i$). This chapter tends to use the representation of δ_i: $S \rightarrow A_i$ such that a ***stationary joint policy*** for these agents can be defined as δ: $S \rightarrow A_1 \times ... \times A_n$. Note δ is equivalent to $(\delta_1, \delta_2,\ldots,\delta_n)$.

Definition 3: An agent i has ***full observability*** if it can observe the global system state. Otherwise, it has ***restricted observability*** where there exists a mapping function G_i: S

$\rightarrow Z_i$. It is assumed that only these two types of observability exist within an *n*-agent DEC-ROMDP.

Definition 4: Given an *n*-agent DEC-ROMDP and a current joint policy $\delta = (\delta_1, \delta_2 \ldots \delta_n)$, the ***steady state probabilities*** can be defined as $x^{\delta} = (x^{\delta}_1, x^{\delta}_2, ..., x^{\delta}_{|S|})$, where x^{δ}_k represents the long run probability that the system state is *k* and $|S|$ is the cardinality of set *S*.

Definition 5: Given an *n*-agent DEC-ROMDP and a current joint policy $\delta = (\delta_1, \delta_2 \ldots \delta_n)$, the ***relative values*** can be defined as $v^{\delta} = (v^{\delta}_1, ..., v^{\delta}_{|S|})$ where

$$g^{\delta} + v^{\delta}_s = r^{\delta}_s + \sum_{s' \in S} p^{\delta}_{ss'} \cdot v^{\delta}_{s'} \quad \forall s \in S .$$

Refer to Howard (1960) for more detail on relative values.

Definition 6: Given an *n*-agent DEC-ROMDP and a current joint policy $\delta = (\delta_1, \delta_2 \ldots \delta_n)$, the associated ***expected reward*** is defined as $\Phi(\delta)$, which is also called the ***gain***, denoted as g^{δ}.

Definition 7: Given an *n*-agent DEC-ROMDP and a current joint policy $\delta = (\delta_1, \delta_2 \ldots \delta_n)$, the following operation is called an individual ***policy update*** by agent *i*.

- If agent *i* has full observability, find a new individual policy δ_i' which satisfies

$$\delta_i'(s) = \arg\max_{a_i \in A_i} (r_s^{(\delta_1(s),\ldots,\delta_{i-1}(s),a_i,\delta_{i+1}(s),\ldots,\delta_n(s))} + \sum_{\forall s' \in S} p_{s,s'}^{(\delta_1(s),\ldots,\delta_{i-1}(s),a_i,\delta_{i+1}(s),\ldots,\delta_n(s))} \cdot v^{\delta}_{s'})$$

  $\forall s \in S$.

- If agent *i* has restricted observability, find a new individual policy δ_i' which satisfies

$$\delta_i'(s) = \arg\max_{a_i \in A_i,} [\sum_{\substack{\forall s \in S \\ G_i(s) = z_i}} x^{\delta}_s (r_s^{(\delta_1(s),\ldots,\delta_{i-1}(s),a_i,\delta_{i+1}(s),\ldots,\delta_n(s))} + \sum_{\forall s' \in S} p_{s,s'}^{(\delta_1(s),\ldots,\delta_{i-1}(s),a_i,\delta_{i+1}(s),\ldots,\delta_n(s))} \cdot v^{\delta}_{s'})]$$
$$\forall s \in S, \; G_i(s) = z_i$$

  $\forall z_i \in Z_i$.

Note the above update keeps individual policies unchanged for every agent except agent *i*, i.e. the new joint policy after the update is $\delta' = (\delta_1, ..., \delta_{i-1}, \delta_i', \delta_{i+1}, ..., \delta_n)$.

Lemma 1: Given an *n*-agent DEC-ROMDP and a current joint policy $\delta = (\delta_1, \delta_2 \ldots \delta_n)$, after a policy update by agent *i*, the joint policy becomes $\delta' = (\delta_1, ..., \delta_{i-1}, \delta_i', \delta_{i+1}, ..., \delta_n)$.

- If agent *i* has full observability, it is guaranteed that $\Phi(\delta') \geq \Phi(\delta)$.
- If agent *i* has restricted observability, it is not guaranteed that $\Phi(\delta') \geq \Phi(\delta)$.

Proof: If agent *i* has full observability, the agent faces a COMDP problem by fixing the other agents' policies. A policy update can be treated as a policy improvement step in Howard's (1960) procedure which guarantees $\Phi(\delta') \geq \Phi(\delta)$. If agent *i* has restricted observability, the agent faces a ROMDP problem by fixing other agents' policies. A policy update can be treated as a policy improvement step in the heuristic algorithm for solving a ROMDP problem. According to Chapter 2, this does not guarantee $\Phi(\delta') \geq \Phi(\delta)$. However, if this happens, the agent has found a local optimum for that ROMDP problem. (Q.E.D)

Definition 8: A *policy update* by agent *i* from $\delta = (\delta_1, \delta_2 \ldots \delta_n)$, to $\delta' = (\delta_1, ..., \delta_{i-1}, \delta_i', \delta_{i+1}, ..., \delta_n)$ is called a ***policy improvement*** if $\Phi(\delta') \geq \Phi(\delta)$.

Definition 9: A joint policy $\delta = (\delta_1, \delta_2 \ldots \delta_n)$ is called ***local optimal policy*** if no policy improvement exists from any agent while fixing the other agents' individual policies. The gain associated with the local optimal policy is called ***local optimal gain***.

Definition 10: A joint policy $\delta = (\delta_1, \delta_2 \ldots \delta_n)$ is called a ***joint myopic policy*** if $\delta(s) = \arg\max_{a \in A} r_s^a, \forall s \in S$. That is, a joint myopic policy chooses an action which maximizes the immediate expected reward for each state.

3.3 DEC-ROMDP Algorithm

This chapter introduces an evolutionary coordination algorithm that updates one agent's policy while keeping other agents' policies unchanged. There exist two variations of the algorithm. The first updates one agent's policy only once and then performs a policy update on the next agent. The second keeps updating one agent's policy until no improvement can be made before performing a policy update on the next agent. Both terminate when no policy update is available at any agent. The details of the algorithms are as follows.

Algorithm I

δ←Initialize joint policy, the gain $g^{\delta} \leftarrow \Phi(\delta)$, and *fail* ←0
while *fail* < *n* **do**
    for *i* = 1 to *n*
        policy update from $\delta = (\delta_1, \delta_2 \ldots \delta_n)$ to $\delta' = (\delta_1, ..., \delta_{i-1}, \delta_i', \delta_{i+1}, ..., \delta_n)$
        if this *policy update* is a *policy improvement* **then**
            $g^{\delta} \leftarrow \Phi(\delta')$, $\delta \leftarrow \delta'$, *fail* ←0
        else
            fail← *fail*+1
        if *fail* = *n* **then**
            break
return g^{δ} and δ.

Algorithm II

δ ← Initialize joint policy, the gain $g^{\delta} \leftarrow \Phi(\delta)$, and *fail* ← 0
while *fail* < *n* **do**
    for *i* = 1 to *n*
        improved ← 0
        while true **do**
            policy update from $\delta = (\delta_1, \delta_2 \ldots \delta_n)$ to
        $\delta' = (\delta_1, ..., \delta_{i-1}, \delta_i', \delta_{i+1}, ..., \delta_n)$
            if this *policy update* is a *policy improvement* **then**
                $g^{\delta} \leftarrow \Phi(\delta')$, $\delta \leftarrow \delta'$, *improved* ← 1
            else
                break
        if *improved* = 1 **then**
            fail ← 0
        else
            fail ← *fail*+1
        if *fail* = *n* **then**
            break
return g^{δ} and δ.

Theorem 1: The above algorithms monotonically increase expect reward, and eventually will terminate at a local optimal policy after a finite number of iterations.

Proof: Both of the algorithms perform a policy update on an agent. If this policy update does not improve the current policy, the next agent is selected to perform the policy update. Hence, the policy is monotonically increasing. As the expected reward is assumed bounded, the algorithm eventually terminates after a finite number of iterations. According to Definition 9, it terminates at a local optimal policy (Q.E.D.)

3.4 Case Study: A Two-Agent DEC-ROMDP problem

3.4.1 General two-agent DEC-ROMDP Models

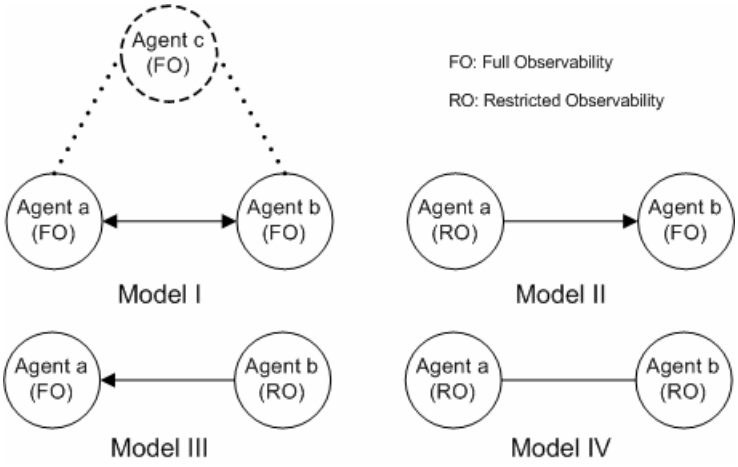

Figure 3.1: General Two-Agent DEC-ROMDP Models with Different Observations

Obviously, several rules exist that may affect the algorithm performance, including the choice of an initial agent for policy update, the policy updating sequence for the agents, and the initial joint policy. This chapter studies two-agent (agent *a* and agent *b*) DEC-ROMDP models (refer to Figure 3.1), so the policy updating sequences can be ignored. Considering the choice of an initial agent for policy update and two variations of the evolutionary coordination algorithm, four approaches are used as follows. "*ab*" ("*ba*") denotes using Algorithm I but the initial agent for updating policy is agent *a* (*b*). Similarly, "*opt_ab*" ("*opt_ba*") denotes using Algorithm II, but the initial agent for updating policy is agent *a* (*b*). In order to compare the approaches "*ab*", "*ba*", "*opt_ba*", and "*opt_ab*", the same myopic policy is initialized for each approach. The term "*meta*" is used to denote the method that chooses the best result from these four approaches.

Additional notation is defined for this two-agent DEC-ROMDP as below.

- $N = |S|$, the number of system states.
- $M_x = |A_x|$, the number of individual actions for agent x.
- $K_x = |Z_x|$, the number of possible observations by agent x.
- J_x: the number of states in each observation set for agent x.

Without loss of generality, assume each observation by an agent corresponds to the same number of states. (N, M_a x M_b, K_a x K_b, J_a x J_b) sufficiently and necessarily

characterizes a two-stage DEC-ROMDP problem. Due to the mapping function G_1 (G_2), $N = K_a$ x J_a (or $N = K_b$ x J_b) always holds. In addition, a fully observable agent can always track the system state such that the mapping function between an observation and a state becomes a one-to-one relationship. Hence the number of states in an observation set by this agent is 1. In Model I, $J_a = 1$ and $J_b = 1$. In Model II, only $J_b = 1$. In Model III, only $J_a = 1$. In Model IV, neither J_a nor J_b equals 1.

To evaluate the effectiveness of the approaches, 1000 instances of different sizes of general DEC-ROMDP problems are generated randomly. In Model I, due to the full observability of agents *a* and *b*, a centralized agent *c* can be assumed to optimize the system with full observability. Obviously, since agent *c* faces a COMDP problem, Howard's (1960) procedure can be used to find optimal policies. It should be noted, however, that experimentation using the evolutionary coordination algorithm shows that the procedure is able to solve a large percentage of problems tested, and for those not solved, the maximum deviation from the optimal gain is negligible (Figure 3.2). Table 3.1 shows that evolutionary coordination algorithm is faster than Howard's procedure for larger Model I problems. As the problem size increases, the execution time does not increase as fast as the Howard's procedure. The reason can be explained as follows. Policy improvement becomes significantly time-consuming due to more states and actions that must be evaluated as the problem sizes increases. A decentralized agent has fewer states and actions to evaluate, thus spends less time improving its local solution than a centralized agent. This is also an advantageous characteristic of solving problems from a decentralized perspective.

For Models II, III and IV, a restricted observable agent is assumed, and the centralized agent is not available to calculate the optimal solution. Brute-force enumeration is used to find the optimal solution in order to evaluate the effectiveness of the evolutionary coordination approach.

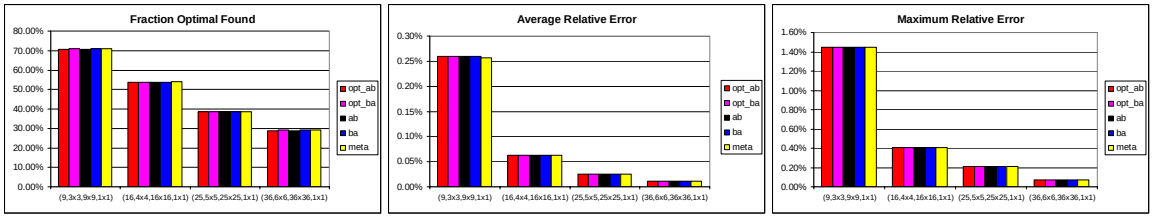

Figure 3.2: Performance for the General Model I Starting with a Joint Myopic Policy

Table 3.1: Average Execution Time (Seconds6) (General Model I)

N, M_axM_b, K_axK_b, J_axJ_b	Howard	opt_ab	opt_ba	a_b	b_a
9,3x3,9x9,1x1	0.0002	0.0006	0.0006	0.0004	0.0004
36,6x6,36x36,1x1	0.0050	0.0139	0.0138	0.0093	0.0124
81,9x9,81x81,1x1	0.1235	0.0238	0.0237	0.0234	0.0235
144,12x12,144x144,1x1	0.2965	0.1155	0.1142	0.0939	0.0960
225,15x15,225x225,1x1	2.6126	0.3060	0.3065	0.2949	0.3026
324,18x18,324x324,1x1	4.6951	0.8654	0.8289	0.8542	0.8667

A joint myopic policy proves effective as a starting policy in finding optimal solutions for Model I. However, the joint myopic policy cannot be used directly to solve Models II, III and IV, since at least one agent is not fully observable. Therefore, we use a randomly generated policy as an initial policy. Obviously, a multiple starts strategy helps offset the disadvantage of randomly choosing a poor initial policy.

Policy perturbation can also be applied to improve the algorithm. The evolutionary coordination algorithm obtains a local optimal policy for Models II, III and IV. Policy perturbation moves an existing local optimum to a neighboring policy by modifying both agents' policy vectors simultaneously. By starting from this neighboring policy, the evolutionary coordination algorithm may come up with a better policy. Combined with multiple starts, policy perturbation works very effectively.

A two phase perturbation strategy is developed that generates neighboring policies by changing policy vector entries of each agent simultaneously. These new policies are then evaluated using the evolutionary coordination algorithms. Before describing the details of the perturbation strategy, define a state aggregation as all states that map to a given observation (z_i) by an agent (i), i.e. $\{s : G_i(s) = z_i\}$.

In the first phase, the second best action (from the final policy improvement phase of Algorithm I or II) for the policy entry of each state aggregation for each agent is evaluated using either evolutionary coordination algorithm I or II. The second phase evaluates adjacent pairs of entries in the policy vector (corresponding to two adjacent state aggregations) for each agent. As in the first phase, the second best alternatives are considered.

[6] The experiments were performed on a computer with Intel Pentium 2.2GHz CPU.

Experimentation indicates that the performance of "*ab*", "*ba*", "*opt_ab*" and "*opt_ba*" are not statistically significantly different. Hence, the following figures were drawn by "*meta*" solutions from different initial starting policies.

As with Model I, this chapter focuses on relative errors when examining results for Models II (Figure 3.3), III (Figure 3.4) and IV (Figure 3.5). For all models, as the problem size increases, the relative error generally decreases. Algorithm performance for Models II and III shows a similar pattern with the average error for all problems less than 0.5% and maximum error of less than 2% after only 20 restarts. Without perturbation, Model IV problems of size (36, 3x3, 6x6, 6x6) have an average error of only 0.38% after 20 multiple starts. With perturbation, this error becomes zero after only 14 restarts, i.e., all 1000 problem instances are solved optimally. For smaller size problems, fewer restarts are needed.

Clearly, as the number of restarts increases, the number of scenarios solved (Percent Optimal Found) is monotonically non-decreasing and the Maximum Relative Error is monotonically non-increasing. However, it should be noted that the Average Relative Error is computed only for those scenarios that are not solved optimally. Thus, the Average Relative Error may not necessarily be monotonically non-increasing since as the number of restarts increase, some scenarios with small relative error may now be solved and no longer contribute to this measure.

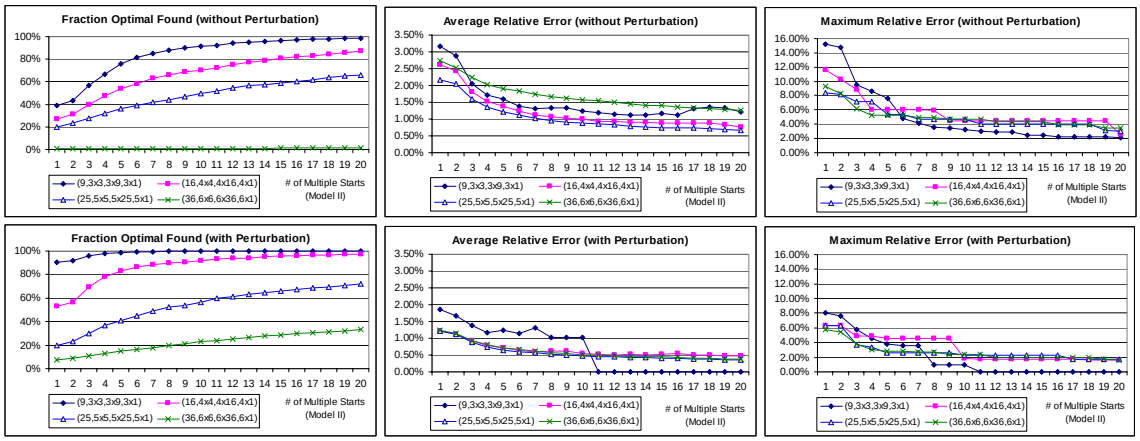

Figure 3.3: Performance for General Model II with Multiple Starts

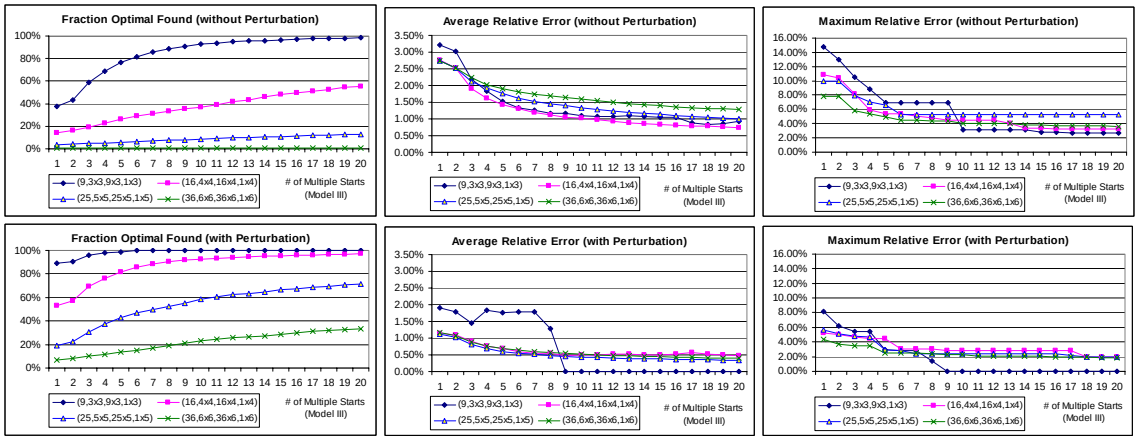

Figure 3.4: Performance for General Model III with Multiple Starts

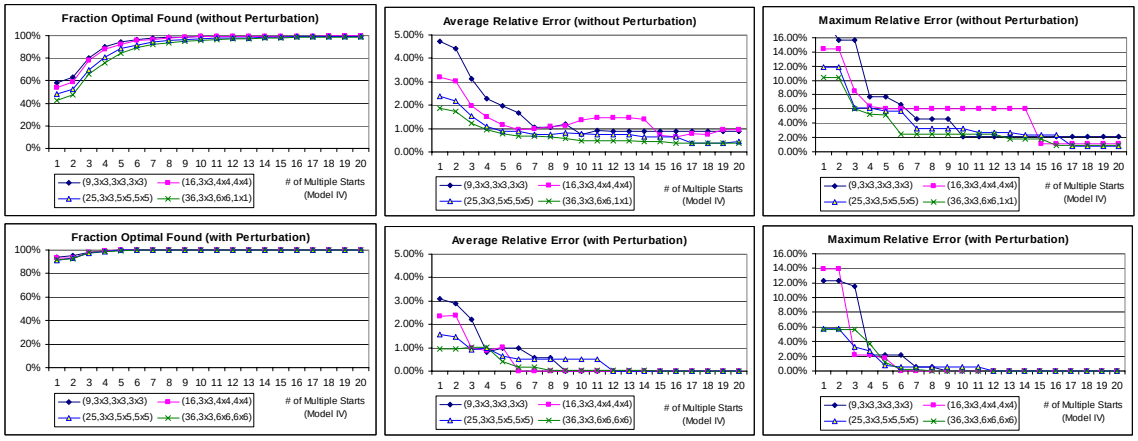

Figure 3.5: Performance for General Model IV with Multiple Starts

3.4.2 DEC-ROMDP Application to a Supply Chain Problem

Next apply the algorithm to a supply chain problem involving a single-product, two-stage supply chain with a supplier and a retailer. The assumptions are as follows. The retailer attempts to satisfy all customer demand from its inventory. Demand each period is independent and identically distributed. The supplier's production and the retailer's order shipment are synchronous, and their lead-time is one period. Each has a maximum inventory capacity. The supplier's production capacity is limited by its inventory capacity, as it cannot produce more than can be accommodated in its warehouse. Likewise, the retailer cannot order more than can be accommodated in its warehouse.

The event sequence of the retailer is described as follows. At the beginning of a period, the retailer's inventory holding cost is incurred. As an order is placed, variable purchase cost and order setup cost are incurred. Note that the purchase cost and setup

cost can be zero if the supplier has zero stock since the supplier can only fill an order from stock. During the period, customer demand occurs and is satisfied from the retailer's stock. Excess demand is lost if there is insufficient stock, and a penalty cost is incurred. The retailer receives the product shipment from the supplier at the end of the period.

The event sequence for the supplier is described as follows. At the beginning of a period, the supplier's inventory holding cost is incurred. As a production order is placed, variable production cost and production setup cost is incurred unless there is no production. Note that the production capacity is limited by the inventory capacity. If the supplier knows retailer's order quantity, he would possibly produce more products based upon this knowledge (this only happens if the retailer shares its inventory information). Subsequently, the supplier receives the retailer's order and ships product from stock. Excess demand is lost. However, there is no penalty cost as this is an interior product transfer in the chain. At the end of the period, the supplier's production is added to inventory.

The gain of the system is long term profit per period for the supply chain. The Markov decision process modeling for this problem is straightforward. An individual inventory level represents a local state, while a system state can be represented by a vector of both inventory levels. Similarly, an individual order quantity represents a local action, while a system action can be represented by a vector of both order quantities.

Parameters for the supply chain are defined as follows.

x: A supply chain member, either the supplier (*sp*) or the retailer (*rt*).

C_x: The inventory capacity for supply chain member x.

W_x: The unit purchase (wholesale) cost for supply chain member x.

H_x: The unit holding cost per period for supply chain member x.

F_x: The setup (fixed) cost per order for supply chain member x.

L_x: The unit stock out (lost sales) cost for supply chain member x.

i_x : The inventory level of supply chain member x, $i_x = 0,1,...,C_x$.

k_x : The production order quantity placed by supply chain member x.

V: The unit selling price to the customer.

d: Demand from the customer, $d = 0,1...D$, assuming $D = C_r$.

The system state *s* is defined as $s = (i_{sp}, i_{rt})$, and a system action *a* is defined as $a = (k_{sp}, k_{rt})$. A system state transition from *s* to s' depends on the corresponding local state transitions $i_{sp} \to i_{sp}'$ and $i_{rt} \to i_{rt}'$ given local actions k_{sp} and k_{rt}, and demand *d*, where $i_{sp}' = (i_{sp} - k_{rt})^+ + k_{sp}$, and $i_{rt}' = (i_{rt} - d)^+ + \min(i_{sp}, k_{rt})$. (Note: $(x)^+ = \max(x,0)$). The corresponding transition probability $p_{ss'}^a$ is derived directly from the demand probability mass function, denoted as $p(d)$. Thus, following system action *a* when in system state *s* and having demand *d* occur implies a transition from state *s* to s'.

In addition, given a system transition from state *s* under action *a* when demand *d* occurs, the retailer's profit during the period, $P_{rt}(s,a,d)$, is calculated as

$$P_{rt}(s,a,d) = V * \min(d, i_{rt}) - [H_{rt} * i_{rt} + W_{rt} * \min(k_{rt}, i_{sp}) + \min(1, \min(k_{rt}, i_{sp})) * F_{rt} + L_{rt} * (d - i_{rt})^+]$$

and the supplier's profit during the period, $P_{sp}(s,a,d)$, is calculated as

$$P_{sp}(s,a,d) = W_{rt} * \min(k_{rt}, i_{sp}) - [H_{sp} * i_{sp} + W_{sp} * k_{sp} + \min(1, k_{sp}) * F_{sp} + L_{sp} * (k_{rt} - i_{sp})^+]$$

The profit for the supply chain when the system transitions from state *s* under system action *a* with demand *d* occurring, i.e. $r_s^a(d)$, is the summation of the retailer and supplier's profits[7], i.e.

$$\begin{aligned} r_s^a(d) &= P_{rt}(s,a,d) + P_{sp}(s,a,d) \\ &= V * \min(d, i_{rt}) - [H_{rt} * i_{rt} + \min(1, \min(k_{rt}, i_{sp})) * F_{rt} + L_{rt} * (d - i_{rt})^+ + H_{sp} * i_{sp} \\ &\quad + W_{sp} * k_{sp} + \min(1, k_{sp}) * F_{sp} + L_{sp} * (k_{rt} - i_{sp})^+] \end{aligned}$$

Thus, r_s^a, the expected immediate reward for state *s* under action *a*, is calculated as

$$r_s^a = \sum_{d=0}^{D} p(d) * r_s^a(d).$$

The following experiments set H_{sp}=1, H_{rt}=1, L_{sp}=0, L_{rt}=100, F_{sp}=40, F_{rt}=0, W_{sp}=10, W_{rt}=50, and V=100. Note that C_{sp} and C_{rt} determine the size of a problem.

This chapter studies four types of information sharing schemes (Figure 3.6) where the critical shared information is the individual inventory levels. The objective is to find optimal policies for the supply chain for each of the different information sharing schemes. In each scheme, supplier and retailer cooperate to maximize profit in the

[7] Notice the total profit does not depend upon W_{rt}, which determines the profit allocation between the supplier and the retailer.

supply chain. In Model I, both supplier and retailer share information. In Model II, the supplier does not know retailer's inventory level. In Model III, the retailer does not know supplier's inventory level. In Model IV, neither supplier nor retailer knows each other's inventory level. This mirrors the four models described above in the general problems. The evolutionary coordination algorithms for DEC-ROMDP problems are used to solve them.

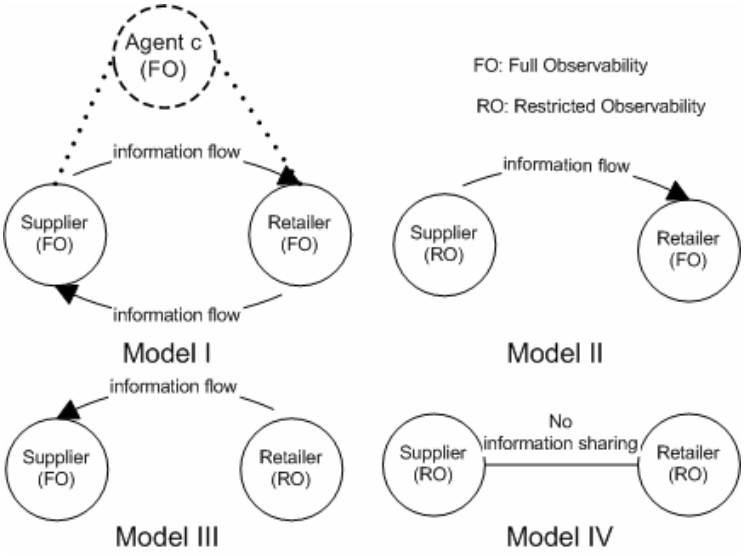

Figure 3.6: Different Information Sharing Schemes for Supply Chain Problems

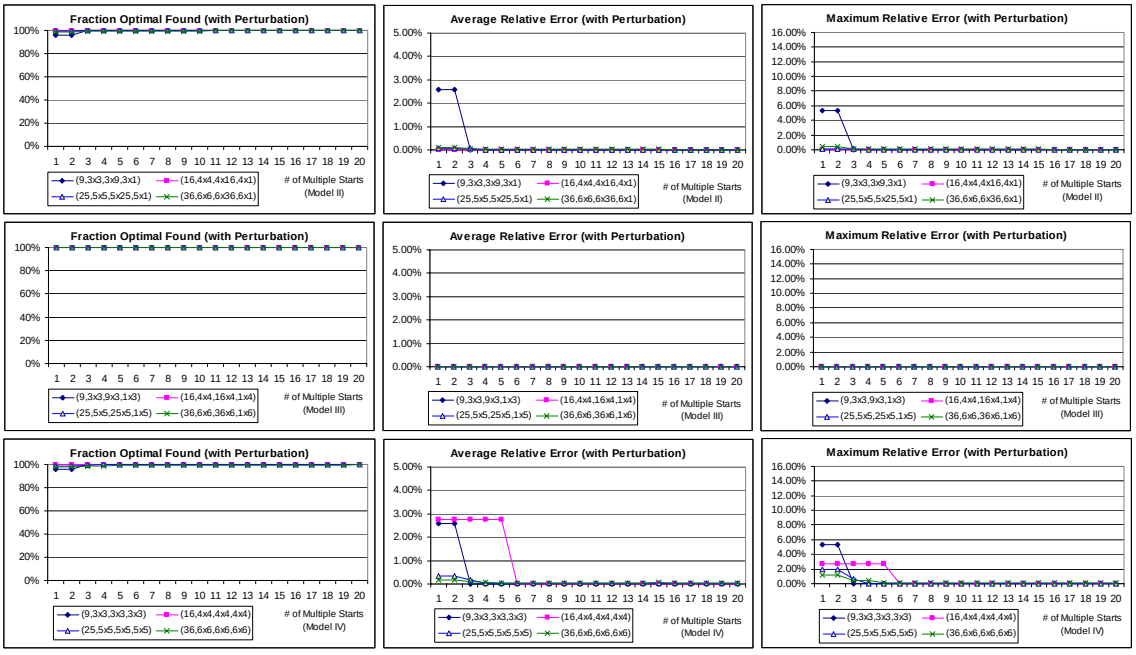

Figure 3.7: Solving Supply Chain Problems with Multiple Starts and Perturbation

Since Model I can always be solved using a centralized agent, the focus is on the other three models. Our computational studies show that the evolutionary coordination algorithm performs even better in supply chain problems than the general problems. For problems with N ranging up to 36, 1000 instances were tested (Figure 3.7). By using a multiple start strategy with perturbations, all problems are solved optimally for Models II and III after only about 4 restarts. For Model IV, after 5 restarts only 2 instances out of 1000 instances for problem (25,5x5,5x5,5x5) and 3 instances out of 1000 instances for problem (36,6x6,6x6,6x6) are not solved optimally. However, the average error for the unsolved problems is only 0.03% and the maximum error is only 0.08% which is probably within the resolution of typical "real" input data.

3.5 Conclusion

Experimental results demonstrate that the *evolutionary coordination* algorithm, coupled with a *multiple start* strategy and *policy perturbation* is effective and efficient for solving DEC-ROMDP problems (especially supply chain problems). The effectiveness of the algorithm is also confirmed for three-stage supply chain problems (Chapter 4). For large scale problems, in order to maintain the efficiency of the algorithm, *successive approximations* (White, 1960) and *encapsulation evolution* approach (Ding *et al*., 1988) can be used to reduce computational effort.

3.6 References

Becker, R., S. Zilberstein, V. Lesser, and C.V. Goldman (2002). Transition-independent decentralized Markov decision processes. In University of Massachusetts/Amherst Computer Science Technical Report, Number 02-50.

Bernstein, D., S. Zilberstein, and N. Immerman (2000). The complexity of decentralized control of MDPs. In Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence.

Boutilier, C. (1999). Sequential optimality and coordination in multiagent systems. Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, 478-485, Stockholm, Sweden.

Chades, I., B. Scherrer, and F. Charpillet (2002). A heuristic approach for solving decentralized-POMDP: Assessment on the pursuit problem. In Proceedings of the Sixteenth ACM Symposium on Applied Computing.

Ding, F.Y., T.J. Hodgson and R.E.King. 1988. A methodology for computation reduction for specially structured large scale Markov decision problems. *European Journal of Operational Research*. 34:105-112.

Howard, R. (1960). Dynamic Programming and Markov Processes. MIT Press, Cambridge, MA

Nair, R., M. Tambe, M. Yokoo, D. Panadath, and S. Marsella (2003). Taming decentralized POMDPs: Towards efficient policy computation for multiagent settings. 18th International Joint Conference on Artificial Intelligence (IJCAI-2003).

Puterman, M.L. (1994). Markov Decision Process: Discrete Stochastic Dynamic Programming, J. Wiley & Sons, New York.

Pynadath, D., and M. Tambe (2002). The communicative multiagent team decision problem: Analyzing teamwork theories and models. *Journal of Artificial Intelligence Research*, 16, 389-423.

Serin, Y. and V.G. Kulkarni (1995). Implementable policies: discounted cost case. in W.J. Steward (Ed.), Computations with Markov Chains. Kluwer Academic Publishers, Dordrecht.

Smith, J.L. (1971). Markov decisions on a partitioned state space. *IEEE Transactions on Systems*, *Man and Cybernetics SMC-1*, no.1: 55-60

Sondik, E.J. (1971). The optimal control of partially observable Markov processes. PhD dissertation, Stanford University, Stanford, California.

White, D.J. (1960). Dynamic programming, Markov chains, and method of successive approximations. *Journal of Mathematical Analysis and Application*, 6, 373-376.

Xuan, P., V. Lesser, and S. Zilberstein (2001). Communication decisions in multiagent cooperation. In *Proceedings of Fifth International Conference on Autonomous Agents*.

Chapter 4 Quantifying the Value of Information and Transfer Price Negotiation in a Supply Chain

4.1 Introduction

4.1.1 Background

It has been reported that information sharing is beneficial to a supply chain, especially in reducing the bullwhip effect (Lee *et al*. 1997, 2000, Cachon and Fisher 2000) and supply chain cost (Gavirneni *et al*. 1999, Swaminathan *et al*. 1997, Tan 1999). However, it may not be beneficial to a supply chain if the cost of adopting the inter-organizational information system is too high (Swaminathan *et al*. 1997, Cohen 2000).

When adopting an information sharing policy, the concern is usually which production information to share, how to share it, and how to use it effectively to maximize mutual benefits to the supply chain members (Huang *et al*. 2003). In a *make-to-order* environment, capacity requirements and production capability of upstream members is critical in order to satisfy the delivery and quantity requirements of the customer (D'Amours *et al*. 1999). In a *make-to-stock* environment, demand information is critical to upstream members in order to mitigate the bullwhip effect. Make-to-stock information sharing problems have been extensively studied (Gavirneni *et al*. 1999, Gavirneni 2002, Simchi-Levi and Zhao 2002) from the supplier's point of view. This chapter quantifies the value of sharing inventory information in a make-to-stock environment and optimizes operational control for the entire supply chain. In addition, this chapter provides insight on how profit can be allocated between the supply chain members so that appropriate information sharing contracts can be negotiated.

Markov Decision Process (MDP) models are used to quantify the value of information sharing and characterize the structure of the optimal policy. A Markov model is a natural way to represent a system where information is shared. Based on the supply chain structure being used, the definition of the state space indicates the available information known to the decision-maker at any point in time. A completely observable MDP (COMDP) is used to model the complete information sharing case. The case of no or limited information sharing is modeled as a MDP with restricted observations (ROMDP). The reader is referred to Chapter 2 for the ROMDP algorithm and Chapter 3

for the decentralized ROMDP (DEC-ROMDP) algorithm. There are several advantages for using an MDP to study supply chain information sharing models. No assumption has to be made about the structure of the optimal policy. Previous approaches have typically made one or more assumptions about the structure of the optimal policy. The effect of these assumptions is discussed in the following section. It is easy to analyze information sharing from different vantage points by structuring the costs from the desired view; total supply chain, retailer, or supplier. Steady state performance parameters are easily determined to provide managerial insight on inventory levels, profit and cost. In addition, there have been a number of developments (e.g. White, 1963, Ding et al., 1988, and Hodgson and Wang, 1992) in computationally efficient methods for MDP problems, which can be used solve large scale problems.

4.1.2 Literature Review

Typically, the structure of the retailer's order policy is assumed to be an order-up-to (Zhao and Simchi-Levi 2002, Lee, So, and Tang 2000) or (*s*,*S*) policy (Gavirneni *et al*. 1999). Chen (1998) considers reorder point/order quantity policies to quantify the value of information sharing when echelon versus installation based reorder points is used. Since the supplier is considered to be perfect from the retailer's perspective and all retailer demand is backlogged, the optimal policy of the retailer is not influenced by how the supplier responds and is thus optimized independent of the supplier's action. As a result, existing inventory control results for single stage problems are used to assert the structure of the retailer's policy or prove the optimality of such structures in the information sharing setting (Clark and Scarf, 1962, Federgruen and Zipkin 1986, Kapuscinski and Tayur 1998). Zhao and Simchi-Levi (2002) extend the work of Kapuscinski and Tayur (1998) to prove the optimality of the supplier's inventory control policy under no information sharing. The resulting information sharing analysis is made from the viewpoint and cost structure of the supplier. In contrast, this dissertation considers an imperfect supplier and pursues system-wide optimality. Retailer and supplier policies are jointly optimized. Since the multi-stage information sharing problem is modeled as an MDP, no assumption has to be made relative to the form of the policy of the supply chain partners.

Gavirneni (2002) addresses how the retailer's optimal (*s*, *S*) policy, when set in 'isolation' does not result in the lowest total supply chain cost. The best way for the retailer to make use of the information and achieve system-wide optimality is to modify the way the retailer responds when information is shared. In order to quantify this, Gavirneni must optimize the entire supply chain given the information sharing partnership is occurring. To determine the lowest supply chain cost, he has to evaluate several suboptimal policies for the retailer, while the supplier's information sharing policy is fixed. This is achieved by performing a search over a range of possible control policy (*s*, *S*) values.

One-stage inventory systems have been extensively examined, and policies, such as base-stock policy, (*s*, *S*) policy, X-Y band policy, etc. (Scarf 1960, Wagner 1972, Shaoxiang and Lamberecht 1996), are optimal given certain assumptions. However, the optimal policy for a general multi-stage supply chain is not of a simple structure and can be very difficult to find because the policy depends on system states and structures (Ng, Li, and Chakhlevitch 2001). An MDP model allows relaxation of the 'fixed policy' assumption.

4.2 Information sharing in a 2-stage Supply Chain

4.2.1 Modeling

4.2.1.1 Assumptions

Consider information sharing in a single-product, 2-stage supply chain with a supplier (manufacturer) and a retailer. The assumptions are as follows. The retailer must satisfy an i.i.d. customer demand distribution. The supplier's production and the retailer's order shipment are synchronous, and their lead-time is one period. Each has a maximum inventory capacity. The supplier's production capacity is limited by its inventory capacity, as it cannot produce more than can be accommodated in its warehouse. Likewise, the retailer cannot order more than can be accommodated in its own warehouse. The cost structure includes production setup/ordering cost, inventory holding cost, production and material cost, and stock out penalty cost.

The event sequence of the retailer is as follows. At the beginning of a period, the retailer's inventory holding cost is incurred. If an order is placed, purchase cost and ordering cost are incurred. Note that the purchase cost and setup cost can be zero if the

supplier has zero stock, since the supplier can only fill an order from stock. During the period, customer demand arrives and is satisfied from retail stock. Excess demand is lost if there is insufficient stock, and a stock out penalty cost is incurred. The retailer receives the product shipment from the supplier at the end of the period.

The event sequence for the supplier is as follows. At the beginning of a period, the supplier's inventory holding cost is incurred. As a production order is placed, production and material cost and production setup cost is incurred unless there is no production. Note that production capacity is limited by inventory capacity. If the supplier knows the retailer's order quantity, he could possibly produce more product-based upon this knowledge (this only happens if the retailer shares inventory information). Subsequently, the supplier receives the retailer's order and ships product from stock. Excess demand is lost. However, there is no penalty cost as this is an interior product transfer in the chain. At the end of the period, the supplier's production is added to inventory.

4.2.1.2 Model Parameters

Parameters for the supply chain model are defined as follows.

x: A supply chain member, either the supplier (s) or the retailer (r).

C_x: The inventory capacity for supply chain member x.

W_x: The unit purchase (wholesale) cost for supply chain member x.

H_x: The unit holding cost per period for supply chain member x.

F_x: The setup (fixed) cost per order for supply chain member x.

L_x: The unit stock out (lost sales) cost for supply chain member x.

i_x : The inventory level of supply chain member x, $i_x = 0,1,...,C_x$.

k_x : The production order quantity placed by supply chain member x.

V: The unit selling price to the customer.

d: Demand from the customer, $d = 0,1...D$, assuming $D = C_r$.

4.2.1.3 Markov Decision Process Approach

The MDP approach to modeling this problem is straightforward. An individual inventory level represents a local state, while a system state can be represented by a combination of both inventory levels. Similarly, an individual order quantity represents a

local action, while a system action can be represented by a combination of both order quantities.

The system state i is defined as $i = (i_s, i_r)$, and a system action a is defined as $a = (k_s, k_r)$. A system state transition from *i* to i' depends on the corresponding local state transitions $i_s \to i_s'$ and $i_r \to i_r'$ given local actions k_s and k_r, and demand *d*, where $i_s' = (i_s - k_r)^+ + k_s$, and $i_{rt}' = (i_{rt} - d)^+ + \min(i_{sp}, k_{rt})$ $i_r' = (i_r - d)^+ + \min(i_s, k_r)$. (Note: $(x)^+ = \max(x,0)$). The corresponding transition probability $p_{ii'}^a$ is derived directly from the demand probability mass function, denoted as *p*(*d*). Thus, following system action *a* when in system state *s* and having demand *d* occur implies a transition from state *i* to i'.

In addition, given a system transition from state *i* under action *a* when demand *d* occurs, the retailer's profit during the period, $P_r(i,a,d)$, is calculated as

$$P_r(i,a,d) = V * \min(d, i_r) - [H_r * i_r + W_r * \min(k_r, i_s) + \min(1, \min(k_r, i_s)) * F_r + L_r * (d - i_r)^+]$$

and the supplier's profit during the period, $P_s(i,a,d)$, is calculated as

$$P_s(i,a,d) = W_r * \min(k_r, i_s) - [H_s * i_s + W_s * k_s + \min(1, k_s) * F_s + L_s * (k_r - i_s)^+].$$

The profit for the supply chain when the system transitions from state *i* under system action *a* with demand *d* occurring, i.e. $r_i^a(d)$, is the summation of the retailer and supplier's profits[8], i.e.

$$\begin{aligned} r_i^a(d) &= P_r(i,a,d) + P_s(i,a,d) \\ &= V * \min(d, i_r) - [H_r * i_r + \min(1, \min(k_r, i_s)) * F_r + L_r * (d - i_r)^+ + H_s * i_s \\ &\quad + W_s * k_s + \min(1, k_s) * F_s + L_s * (k_r - i_s)^+] \end{aligned}$$

Thus, r_i^a, the expected immediate reward for state *i* under action *a*, is calculated as

$$r_i^a = \sum_{d=0}^{D} p(d) * r_i^a(d).$$

4.2.1.4 Information Flows

Our objective is to quantify the value of sharing inventory information, and to examine the sensitivity to demand characteristics, production/setup cost, inventory

[8] Notice the total profit does not depend upon W_r, which determines the profit allocation between the supplier and the retailer.

capacity, holding cost, and stock out penalty cost. In order to determine the value of the supplier's or the retailer's inventory information, four types of information sharing models are to be analyzed (see Figure 4.1). Davis's model is included in Figure 4.1.

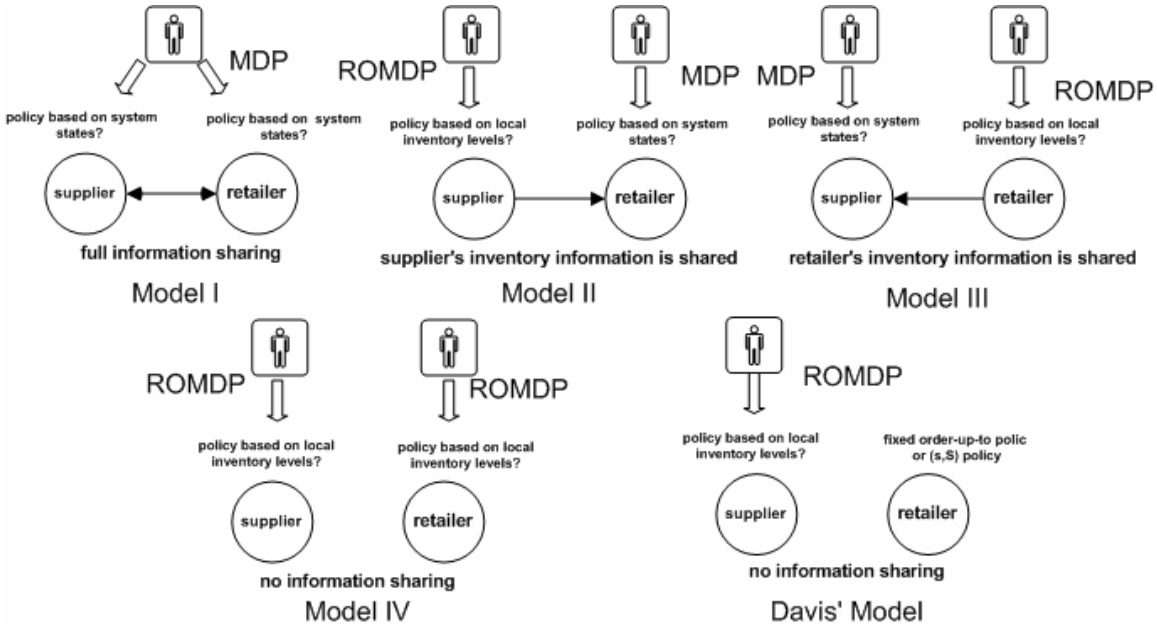

Figure 4.1: Information Flow in Each Model

MDP is used as the modeling approach, with the objective to find optimal joint policies. As both supplier and retailer make their own replenishment decisions collectively to maximize the total profit per period for the supply chain, each model is a decentralized decision problem.

Assume that an agent SA makes decisions for the supplier, and an agent RA makes decision for the retailer. In Model I, supplier and retailer share inventory information with each other. This is a special DEC-ROMDP problem, since both SA and RA see the entire state of the system. Thus, an equivalent centralized agent CA can make decisions for the whole system by solving a single agent MDP problem. Once the centralized decision is determined, decentralized decisions by SA and RA can be easily derived. In Model II, only the supplier's inventory information is shared with the retailer. SA sees only local states while RA sees the entire state of the system. In Model III, only the retailer's inventory information is shared to the supplier. SA sees the entire state of the system while RA sees only local states. In Model IV, there is no information sharing between supplier and retailer. Each agent sees only its local states. Obviously, no centralized agent can be utilized to help make decisions for the decentralized agents

within Models II, III, and IV. These three models are typical DEC-ROMDP problems. In addition, Davis' model (2004) is a single agent ROMDP problem. Details of single agent ROMDP and DEC-ROMDP problems are contained in Chapter 2 and Chapter 3 respectively.

4.2.2 Methodology

To quantify the value of sharing inventory information, four different models need to be considered. Howard's procedure (1960) can be used to solve Model I, while the DEC-ROMDP algorithm (Chapter 3) is used to solve Models II, III, and IV. The DEC-ROMDP algorithm, also called the *evolutionary coordination* algorithm, updates the policy for each agent by turns until no more policy improvements can be made by any agent. During a policy update, all agents' policies are fixed except the updating agent. The policy update differs depending upon whether the updating agent has full observation or restricted observation of the system states. If the agent has full observability, the policy update guarantees an improved gain, or at least an equivalent gain. If the agent has restricted observability, the policy update does not guarantee an improved gain. This evolutionary coordination algorithm eventually terminates after a finite number of iterations. *Policy perturbation* and *multiple starts* are used to improve the algorithm. Experiments demonstrate that the algorithm with policy perturbation and multiple starts optimally solves almost all the supply chain problems attempted, and has negligible error for those problems not solved optimally. Experimental evidence indicates that large-sized supply chain problems can be expected to be solved with, at worse, negligible errors. *Successive Approximation* (White 1960) is used to reduce computational effort. The *encapsulated evolutionary* approach (Ding *et al*. 1985) can be used for further reduction of computational effort (Davis 2004).

Solving the four models enables the quantification of the value of sharing information. The average long term profit per period over the supply chain is the gain of the underlying MDP. Let g_i denote the gain for Model *i*. Obviously, one would expect that $g_1 \geq g_2 \geq g_4$, and $g_1 \geq g_3 \geq g_4$, since more information should always help a decision maker. The Relative Information Value (*RIV*) is of more interest. By using Model IV as a reference, $RIV_1 = (g_1 - g_4)/g_4$ is used to represent the relative value of

shared information between supplier and retailer, $RIV_2 = (g_2 - g_4)/g_4$ to represent the relative value of sharing only the supplier's inventory information with the retailer, and $RIV_3 = (g_3 - g_4)/g_4$ to represent the relative value of sharing only the retailer's inventory information with the supplier.

4.2.3 Experimentation

In the computational experiments, the selling price is set at 100, and all other cost parameters are accordingly based on reasonable assumptions of real world conditions, such as markup ratio and interest rate. Considerations in the experimental design are summarized as follows:

1. The retailer's order cost is assumed to be 0 as this cost is usually overhead in the real world.
2. Since the goal is to optimize the total supply chain, the supplier has no stock out cost.
3. The retailer's purchasing cost can be regarded as the transfer cost between supplier and retailer, which does not affect total profit, but does affect individual profits.
4. The holding cost is based on the cost during an order period (e.g., a week).
5. The markup ratio ranges from 5:1 to 20:1, which determines the supplier's production cost.
6. The demand distribution is assumed to be a truncated discrete Normal distribution. Mean demand is denoted as "Mean". The coefficient of variation (*Cov*), is used to measure the variability of the demand distribution.
7. The retailer's inventory capacity is set such that the probability of a lost sale is not large.
8. In the real world, the supplier may have limited production capacity, which is reflected by its inventory capacity.

Two experimental designs are performed. The first investigates the effect of production capacity, mean demand and the coefficient of variation on the relative value of information. The second investigates the effect of the coefficient of variation, holding cost, the supplier's production setup cost, and the retailer's penalty cost on the relative value of information.

4.2.3.1 Design of Experiments I

The first experimental design (denoted as DOE1) is shown in Table 4.1. 12 problems are generated and solved using the evolutionary coordination algorithm with policy perturbation and 20 restarts. Lost sales[9] for all the problems range from 0.2% to 7%, with most of the lost sales being less than 5%. These percentages appropriately reflect the real world.

Table 4.1: Design of Experiments I

Mean[10]	Cov	C_s	C_r	H_s	H_r	W_s	W_r	L_s	L_r	F_s	F_r	V
5	0.3	11	15	1	1	10	50	0	100	40	0	100
	0.45	13										
	0.60	15										
		17										

For the 12 problem instances, $RIV_1 = RIV_3$, and $RIV_2 = 0$, except $RIV_2 = 0.03\%$ for one problem. This is intuitive since the retailer's decision making is closely related to current inventory levels. Further, the retailer is not concerned with order setup cost as $F_r = 0$.[11] Knowing the supplier's inventory information, the retailer may choose to delay ordering so that the supplier can make fewer production setups. Thus, it is possible to reduce total cost and the supplier's information may be helpful. However, this situation rarely occurs. If there is any value in the supplier sharing information, it appears to be small. The exception of 0.03% confirms the point. The followings only consider the retailer sharing information, i.e., RIV_3.

[9] Lost sale here represents the average number of demand lost per period.
[10] Experiments indicate that a mean demand of 7.5 would not be appropriate, since it causes huge lost sales.
[11] If the retailer's setup cost is not zero, RIV_2 would be relatively larger based on experiments.

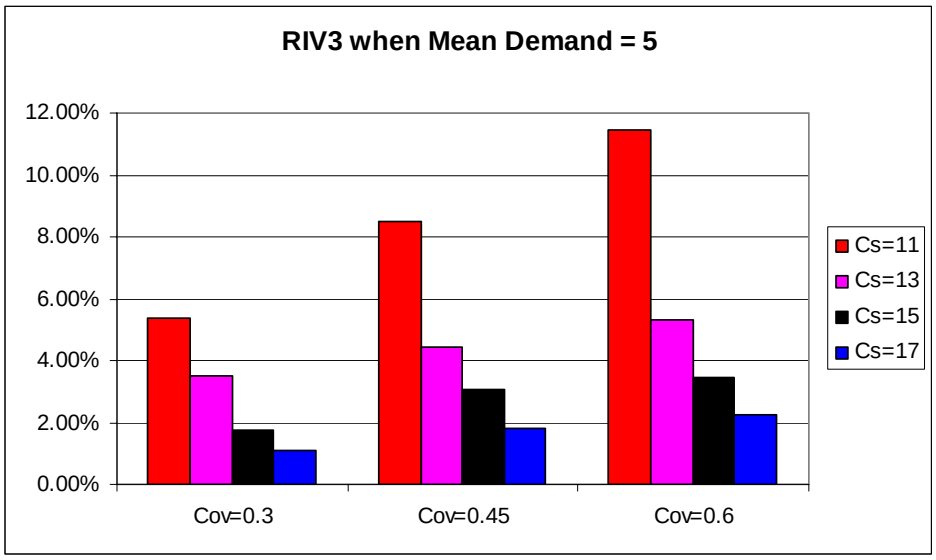

Figure 4.2: Relative Information Value when Mean Demand = 5

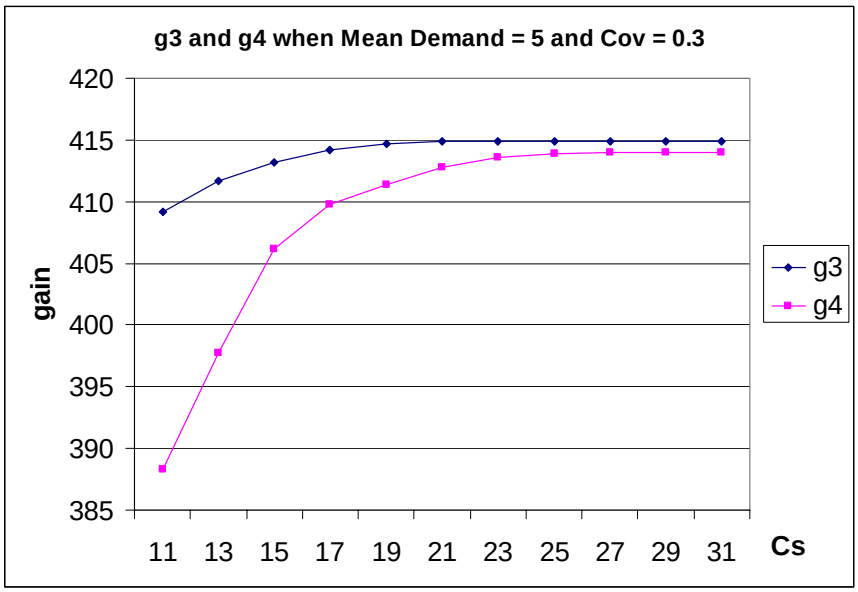

Figure 4.3: g_3 and g_4 vs. Cs, with Mean Demand = 5 and Cov = 0.3

Figure 4.2 indicates that RIV_3 increases as demand variability increases. This is intuitive as a supplier with retailer inventory information may predict retailer demand and make production decisions accordingly. Figure 4.2 also indicates that RIV_3 decreases as the supplier's production capacity increases. As capacity increases, the supplier may produce more during some periods and may not order at all during other periods. Setup cost is reduced and holding cost is increased, which subsequently may reduce penalty cost. Consequently, supply chain profit increases, i.e., g_3 and g_4 increase (e.g. Figure 4.3). There exists a maximum supply chain profit that can be achieved, once customer

demand is assumed. When the retailer's inventory information is shared, the supplier has additional information about the retailer's pending order, but as supplier capacity increases, the value of that information decreases.

Figure 4.3 shows g_3 and g_4 as a function of supplier capacity (C_s) with a mean demand of 5 and *Cov* of 0.30. It is obvious that at some point the incremental value of capacity will go to zero, and that $g_3 > g_4$ always holds ($g_3 - g_4$ becomes a constant as capacity increases, g_3 does not increase for $C_s > 25$, and g_4 does not increase for $C_s > 29$).

4.2.3.2 Design of Experiments II

Considering the results from DOE1, experiment II (DOE2) follows Table 4.2. Thus, 96 problems are generated and solved. Each problem is solved using the evolutionary coordination algorithm with policy perturbation and 20 restarts.

Table 4.2: Design of Experiments II

Mean	*Cov*	C_s	C_r	H_s	H_r	W_s	W_r	L_s	L_r	F_s	F_r	V
5	0.30	15	15	1	1	10	50	0	100	40	0	100
	0.45			2	2	20			200	80		
	0.60											

Effect tests using JMP 5.1 indicate that all the main effects except H_r are significant at the 99% significance level. Among the main effects, *Cov* has the highest effect, then F_s, L_r, H_s, W_s, until H_r, which is almost negligible.

It is not surprising that a larger *Cov* makes information more valuable. This was shown in DOE1. As L_r, F_s or W_s increases, the total cost in the supply chain increases and profit decreases. This partially contributes to the increase of RIV_3. In addition, being able to track the retailer's inventory, the supplier performs better.

The interactions among *Cov*, W_s, L_r and F_s are significant, but the interactions involving H_s or H_r are not significant. Also, the higher a certain factor is, the more significant is the interaction between this factor and another factor. The reason for this is intuitive since a supplier with the retailer's information will perform better in a highly variant environment.

4.3 Information sharing in a 3-stage Supply Chain

4.3.1 Modeling and Methodology

Now study information sharing in a single-product, 3-stage serial supply chain with a capacitated manufacturer, a capacitated supplier and a retailer. Assumptions and event sequences for the 3-stage supply chain are similar to those for a 2-stage supply chain. There is no penalty cost for the supplier or the manufacture as this is an interior product transfer in the chain. In the 2-stage supply chain, "x" is used to denote supplier (s) or retailer (r). In the 3-stage supply chain, "x" is also used to denote manufacturer (m).

Therefore, an individual inventory level represents a local state; while a system state i is represented by a combination of the three members' inventory levels. Similarly, an individual order quantity represents a local action, while a system action a can be represented by a combination of three members' order quantities.

Given a typical period, system state $i = (i_m, i_s, i_r)$, action $a = (k_m, k_s, k_r)$ and demand *d*, the retailer's profit is calculated as

$$P_r(i,a,d) = V \cdot \min(d, i_r) - [H_r \cdot i_r + W_r \cdot \min(k_r, i_s) + \min(1, \min(k_r, i_s)) \cdot F_r + L_r \cdot (d - i_r)^+]$$

The supplier's profit is calculated as

$$P_s(i,a,d) = W_r \cdot \min(k_r, i_s) - [H_s \cdot i_s + W_s \cdot \min(k_s, i_m) + \min(1, \min(k_s, i_m)) \cdot F_s + L_s \cdot (k_r - i_s)^+]$$

The manufacturer's profit is calculated as

$$P_m(i,a,d) = W_s \cdot \min(k_s, i_m) - [H_m \cdot i_m + W_m \cdot k_m + \min(1, k_m) \cdot F_m + L_m \cdot (k_s - i_m)^+]$$

Total profit for the supply chain is the summation of the above three profits[12].

$$\begin{aligned} r_i^a(d) = V \cdot \min(d, i_r) - [&H_r \cdot i_r + \min(1, \min(k_r, i_s)) \cdot F_r + L_r \cdot (d - i_r)^+ \\ &+ H_s \cdot i_s + \min(1, \min(k_s, i_m)) \cdot F_s + L_s \cdot (k_r - i_s)^+ \\ &+ H_m \cdot i_m + W_m \cdot k_m + \min(1, k_m) \cdot F_m + L_m \cdot (k_s - i_m)^+] \end{aligned}$$

Thus, r_i^a, the expected immediate reward given a state *i*, and an action *a*, is calculated as

$$r_i^a = \sum_{d=1}^{D} p(d) \cdot r_i^a(d).$$

[12] Notice the total profit does not depend upon W_r and W_s, which determine the profit allocation along the supply chain members.

From the 2-stage supply chain, it was found that information sharing from an upper stream member to a down stream member is not particularly valuable. Moving to a 3-stage supply chain, this dissertation focuses on up-stream information flow as shown in the following eight models (Figure 4.4).

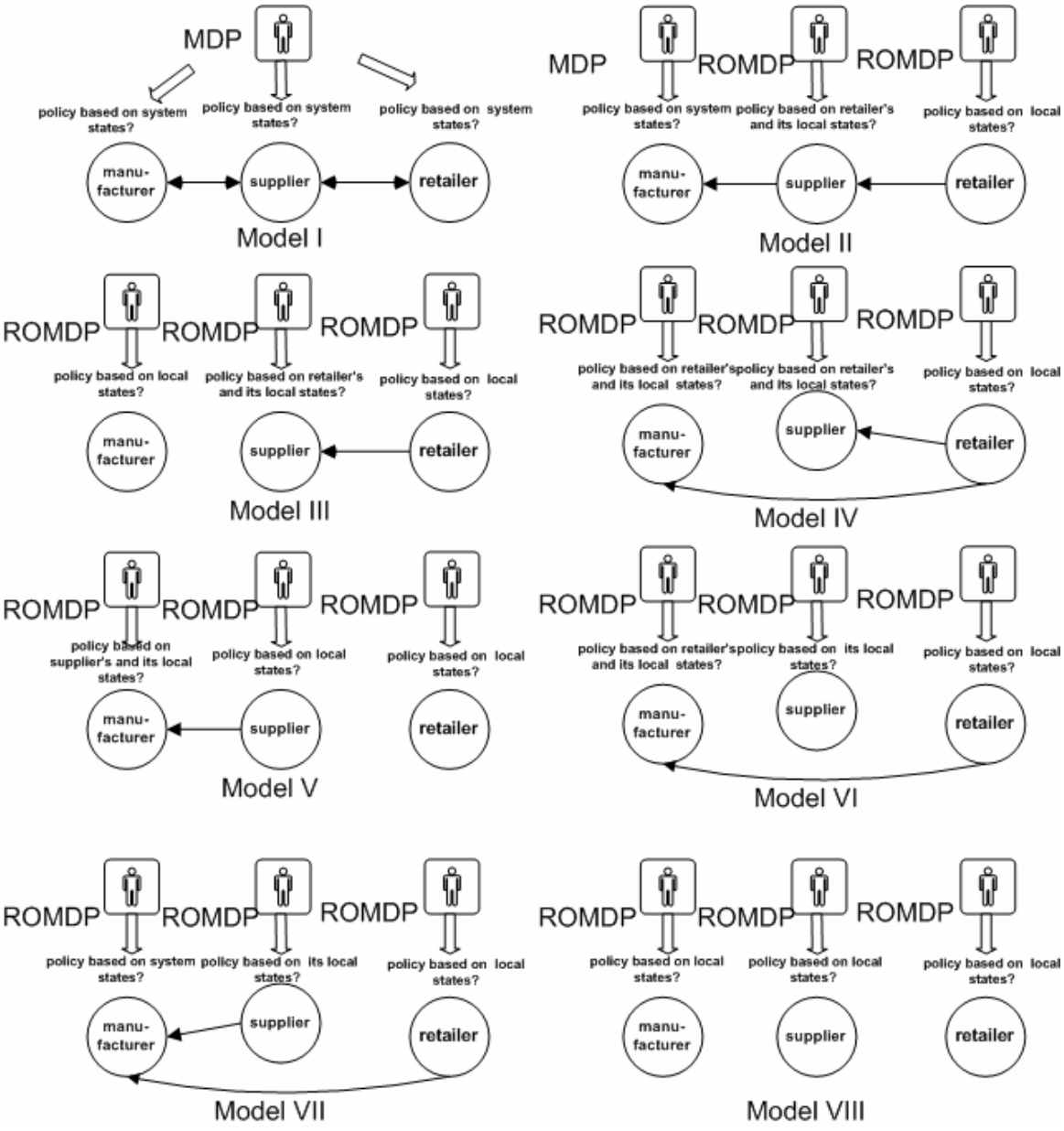

Figure 4.4: Information Flow in Each 3-Stage Model

The objective is to find optimal joint policies for each model. All members make their own replenishment decisions collectively to maximize profit per period for the supply chain. Model I is a single agent MDP problem, while the other models are DEC-ROMDP problems. The methodology for calculating relative information value is the

same as for the 2-stage model. Let g_i denote the gain for Model i. As Model VIII has no information sharing, the relative information value for Model i is defined as $RIV_i = (g_i - g_8)/g_8$.

4.3.2 Experimentation

4.3.2.1 Design of Experiments III

Table 4.3: Design of Experiments III

Mean	Cov	C_m	C_s	C_r	H_m	H_s	H_r	W_m	W_s	W_r	L_m	L_s	L_r	F_m	F_s	F_r	V
3	0.3	10	10	10	1	1	1	5	20	50	0	0	100	40	40	0	100
	0.6				2	2	2	10					200	80	80		

Design of Experiments III (DOE3) is in Table 4.3. 256 problems are generated and solved. Effect tests were made using JMP 5.1. All the main effects (H_m, H_s, H_r, W_m, L_r, F_m, and F_s) are significant for most *RIV*s. Of the 28 interactions only 8 are significant (H_m*F_s, H_m*L_r, H_m**Cov*, W_m*F_s, W_m**Cov*, F_m*F_s, F_s*L_r, F_s**Cov* and L_r**Cov*).

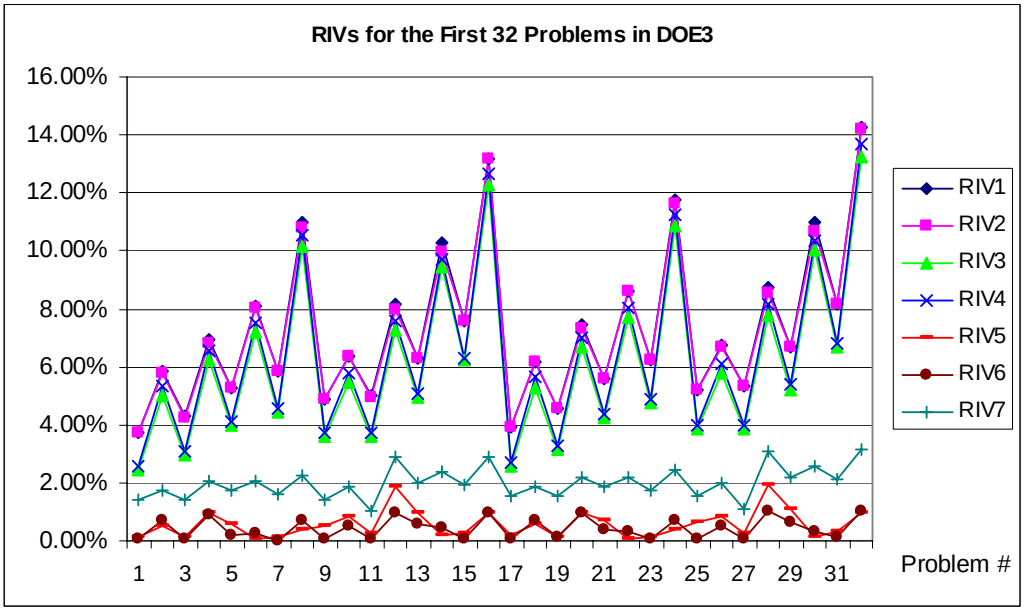

Figure 4.5: RIVs for the First 32 Problems in DOE3 (all holding costs = 1)

Table 4.4: Summary of Mean Comparison for RIVs

Oneway Analysis of *RIV*s by Group for 256 Problems in DOE3							
RIV_i	RIV_1	RIV_2	RIV_3	RIV_4	RIV_5	RIV_6	RIV_7
Mean	0.0776	0.0769	0.0626	0.0582	0.0058	0.0075	0.0272
Tukey-Kramer	A	A	B	B	D	D	C

The *RIV*s can be separated into four groups. Figure 4.5 shows *RIV*s for the first 32 problems in DOE3. It demonstrates visually that *RIV*5 and *RIV*6 comprise a group which gives the least value for information sharing. RIV_7 is the second group. It gives statistically greater values for information sharing. RIV_3 and RIV_4 comprise the third group, while RIV_1 and RIV_2 comprise the most valuable group. Table 4.4 lists the mean for each *RIV* over 256 problems and Tukey-Kramer test result, which confirms the grouping.

The above grouping indicates the following. First, information flow from the retailer to the supplier (s←r) is most valuable, while information flow from the retailer to the manufacturer (m←r), or from the supplier to the manufacturer (m←s) is comparatively less valuable. Second, combining information flows m←r and m←s helps some, but falls well short of the performance of the flow s←r. Third, once information flow s←r has been implemented, a good choice for additional information sharing would be to add flow m←s, rather than flow m←r. What may be implied here is that having information from the immediate successor stage, for every stage, is the most important information to have.

4.3.2.2 Design of Experiments IV

Table 4.5: Design of Experiments IV

Mean	Cov	C_m	C_s	C_r	H_m	H_s	H_r	W_m	W_s	W_r	L_m	L_s	L_r	F_m	F_s	F_r	V
3	0.3	8	8	10	1	1	1	5	20	50	0	0	100	40	40	0	100
	0.6	10	10														
		12	12														

Design of Experiments IV (DOE4) is in Table 4.5. 18 problems are generated and solved. Since the information flow from the retailer to the supplier is most significant, this dissertation focuses on how RIV_3 changes with the capacities of the supplier and manufacturer, and with demand variability. Figure 4.6 shows that RIV_3 decreases as the supplier capacity (C_s) increases, while the manufacturer's capacity (C_m) does not appear to significantly affect RIV_3.

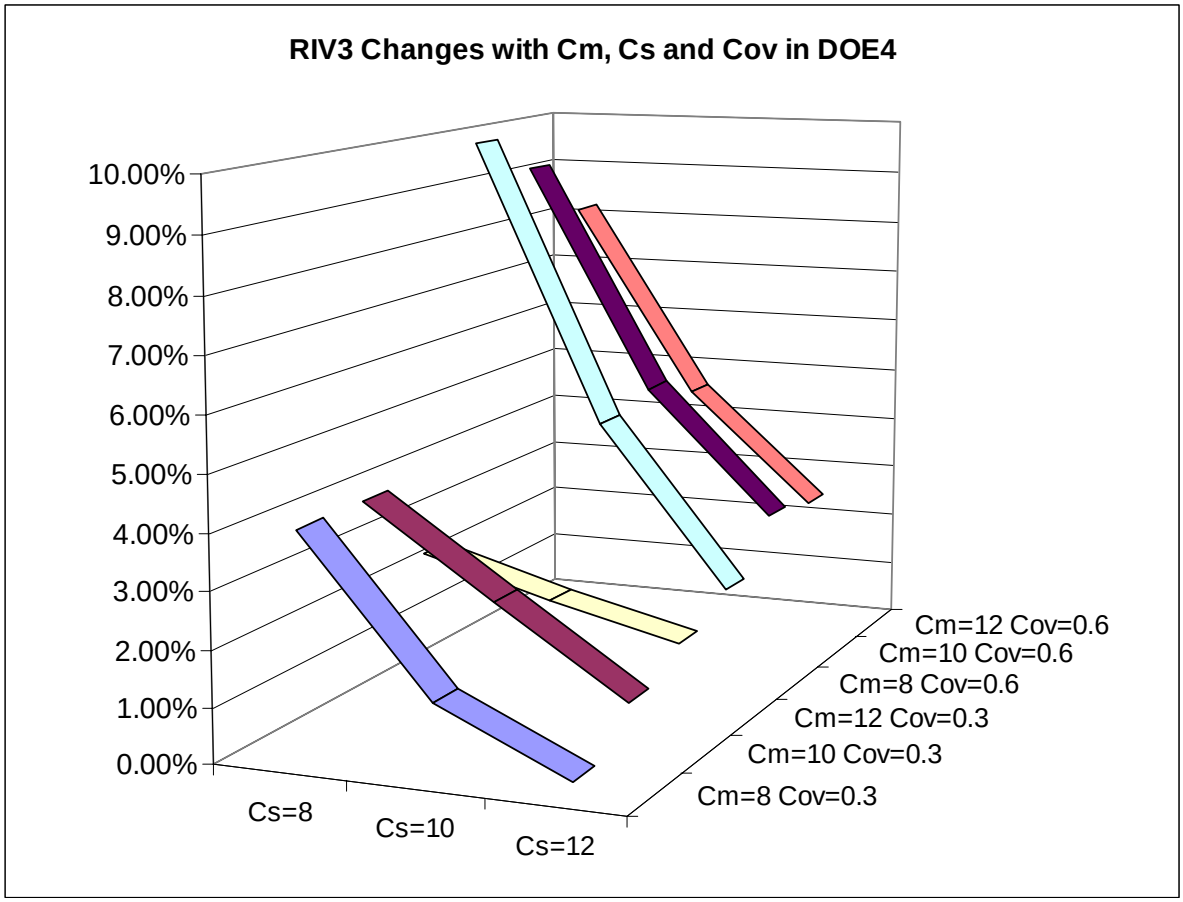

Figure 4.6: RIV_3 Changes with *Cm*, *Cs* and *Cov*

4.4 Transfer Cost Negotiation

By determining the optimal policy for a supply chain (with or without information sharing), the total profit per period within the supply chain is obtained. The question is: how should the additional profit be allocated to the individual supply chain members after information sharing is implemented? In order to achieve equity among the supply chain members, the transfer cost from member to member should be fairly negotiated. Once the transfer cost is determined, the unit purchase cost[13] can then be calculated, since the average order quantity for each member can be determined for a given operating policy. The limits are determined within which the transfer cost negotiation should be conducted so that the needs of every supply chain member are fulfilled, but have not modeled the actual negotiation process. For ease of analysis, it is assumed that profit is positive for all stages and for all models. In what follows, the transfer cost negotiation

for a 2-stage supply chain and a 3-stage supply chain are analyzed. Without loss, it is assumed that the system is taken from no information sharing to some form of information sharing.

4.4.1 Determination of Transfer Cost in a 2-stage Supply Chain

Initially, the cost of information sharing (CI) is ignored in the determination of limits on how a fair transfer cost should be negotiated. This section has only a supplier (manufacturer) and a retailer. Let us define notation for the No Information Sharing (NIS) case.

S_0 retailer's sales revenue per period

C_{0s} supplier's cost per period, including production and materials, inventory holding, and setup cost

C_{0r} retailer's cost per period including inventory holding, and stock out penalty.

$C_0 = C_{0s} + C_{0r}$ (total cost per period in the supply chain)

$P_0 = S_0 - C_0$ (total profit per period in the supply chain)

Z_0 retailer's transfer (purchase) cost per period

$P_{0s} = Z_0 - C_{0s}$ (supplier's profit per period)

$P_{0r} = S_0 - C_{0r} - Z_0$ (retailer's profit per period)

Notation for the Information Sharing (IS) case can be defined in the same way, but replace the subscript "0" with "1." Note that the MDP approach allows straightforward computation of the values S_0, C_{0s}, C_{0r}, P_0, S_1, C_{1s}, C_{1r}, and P_1. Given the initial transfer cost Z_0, the problem is to determine transfer cost Z_1 after implementing information sharing. Without loss, it is assumed that supplier and retailer make a profit both before and after information sharing is implemented. Thus, the following 4 constraints are given.

$$P_{0s} = Z_0 - C_{0s} > 0 \quad (1)$$

$$P_{0r} = S_0 - C_{0r} - Z_0 > 0 \quad (2)$$

$$P_{1s} = Z_1 - C_{1s} > 0 \quad (3)$$

$$P_{1r} = S_1 - C_{1r} - Z_1 > 0 \quad (4)$$

[13] The unit purchase cost is how much that the downstream member pays its immediate upstream member. The transfer cost per period equals the product of the unit purchase cost and the order quantity.

Since (initially) the cost of sharing information is ignored, it is reasonable to assume that both supplier and retailer make more profit after implementing information sharing. Thus,

$$P_{1s} > P_{0s} \quad \rightarrow \quad Z_1 - C_{1s} > Z_0 - C_{0s} \qquad (5)$$

$$P_{1r} > P_{0r} \quad \rightarrow \quad S_1 - C_{1r} - Z_1 > S_0 - C_{0r} - Z_0 \qquad (6)$$

Figure 4.7 is drawn based on constraints (1), (2), (3), (4), (5), and (6). They define the area (parallelogram) within which negotiations must occur. The NIS transfer cost must be within the line segment TT' and the IS transfer cost must be within the line segment WW' so that everyone makes their profit. If the transfer cost before information sharing is $Z_0 = x^*$, U is the point where additional profit is only obtained by the retailer, and U' is the point where additional profit is only obtained by the supplier. If the retailer's average order quantity changes from q_0 (before information sharing) to q_1 (after information sharing), the unit purchase cost W_r should change from x^*/q_0 (NIS) to y^*/q_1 (IS). This provides guidance for price negotiation on the unit purchase cost.

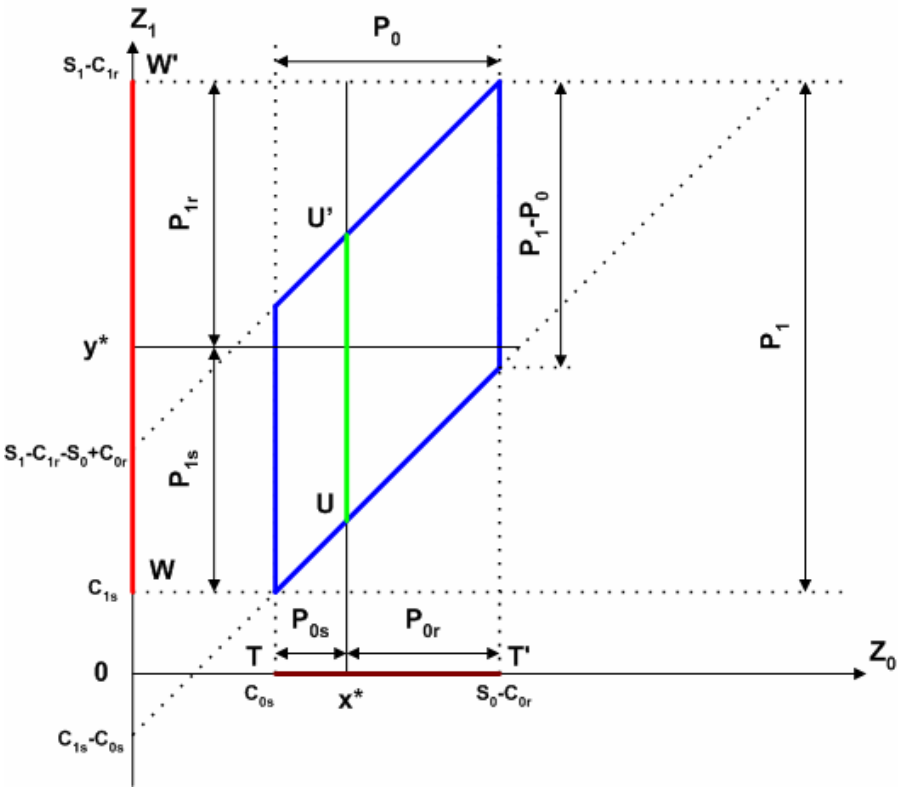

Figure 4.7: Transfer cost between supplier and retailer when CI = 0

If the cost of implementing information sharing is charged to the supplier, then constraints (3) and (5) change to (7) and (8), respectively.

$$P_{1s} = Z_1 - C_{1s} - CI > 0 \quad (7)$$

$$P_{1s} > P_{0s}, \text{ i.e. } Z_1 - C_{1s} - CI > Z_0 - C_{0s} \quad (8)$$

Figure 4.8 is drawn based on the constraints (1), (2), (4), (6), (7) and (8). The parallelogram that defines where the negotiation must occur becomes smaller vertically, and the price negotiation should be conducted correspondingly.

Given x^*, the upper limit and lower limit of y^* can be calculated as:

$$y^*_{max} = S_1 - C_{1r} - S_0 + C_{0r} + x^* \quad (9)$$

$$y^*_{min} = C_{1s} - C_{0s} + x^* + CI \quad (10)$$

Given a NIS transfer cost x^* that defines a profit allocation ratio, if it is desired that supplier and retailer benefit according to the same ratio after information sharing, the IS transfer cost y^* is defined by the intersection of the line segments GG' and $Z_0=x^*$. G is the point where all profit is solely allocated to the retailer and G' is the point where all profit is solely allocated to the supplier, in both the NIS and IS cases.

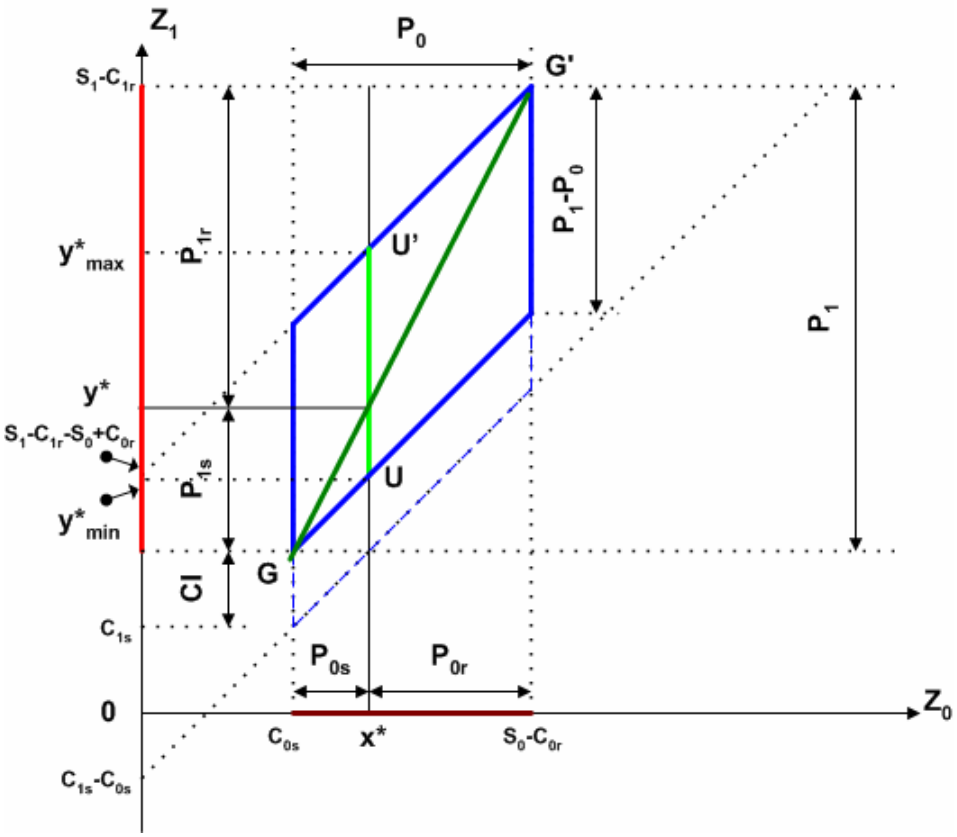

Figure 4.8: Transfer cost between supplier and retailer, when CI is charged to the supplier

If the cost of information sharing (CI) is too great, the parallelogram may not exist and information sharing is not worthwhile. Using the set of 96 problems solved in

DOE2 (See Appendix) as an example, as CI increases information sharing is worthwhile on fewer and fewer problems. If CI is 5.5% or more of gross revenue, information sharing is not beneficial for any problem.

4.4.2 Determination of Transfer Cost in a 3-Stage Supply Chain

What is interesting to note is that a member of the supply chain can possibly benefit from information sharing, yet be uninvolved in either giving or receiving data. For instance, if the retailer shares information only with the supplier, the manufacturer may benefit. This dissertation ignores the politics of this issue. Initially, again ignore the cost of information sharing in the determination of limits in the negotiation of fair transfer costs by supplier and retailer. Again, let us define notation for the NIS case.

S_0 retailer's sales revenue per period.

C_{0m} manufacturer's cost per period, including production and material, inventory holding, and setup.

C_{0s} supplier's cost (except purchase cost) per period, including setup, and inventory holding.

C_{0r} retailer's cost (except purchase cost) per period, including inventory holding and stock out penalty.

Z_{0s} supplier's transfer (purchase) cost per period

Z_{0r} retailer's transfer (purchase) cost per period

$C_0 = C_{0m} + C_{0s} + C_{0r}$ (total cost per period in the supply chain)

$P_0 = S_0 - C_0$ (total profit per period in the supply chain)

$P_{0m} = Z_{0s} - C_{0m}$ (manufacturer's profit per period)

$P_{0s} = Z_{0r} - C_{0s} - Z_{0s}$ (supplier's profit per period)

$P_{0r} = S_0 - C_{0r} - Z_{0r}$ (retailer's profit per period)

Notation for the IS case can be defined as before. Use "1" to replace the subscript "0." Note again that the MDP approach provides a perfect tool to easily obtain values for S_0, C_{0m}, C_{0s}, C_{0r}, P_0, S_1, C_{1m}, C_{1s}, C_{1r}, and P_1. Given the initial transfer costs Z_{0s} and Z_{0r}, the problem is to determine Z_{1s}, and Z_{1r}. Again without loss, it is assumed that manufacturer, supplier and retailer make a profit both before and after information sharing is implemented. The following 6 constraints are given.

$$P_{0m} = Z_{0s} - C_{0m} > 0 \quad (11)$$

$$P_{0s} = Z_{0r} - C_{0s} - Z_{0s} > 0 \quad (12)$$

$$P_{0r} = S_0 - C_{0r} - Z_{0r} > 0 \quad (13)$$

$$P_{1m} = Z_{1s} - C_{1m} > 0 \quad (14)$$

$$P_{1s} = Z_{1r} - C_{1s} - Z_{1s} > 0 \quad (15)$$

$$P_{1r} = S_1 - C_{1r} - Z_{1r} > 0 \quad (16)$$

Assuming every member makes more profit after information sharing,

$$P_{1m} > P_{0m}, \quad \text{i.e. } Z_{1s} - C_{1m} > Z_{0s} - C_{0m} \quad (17)$$

$$P_{1s} > P_{0s}, \quad \text{i.e. } Z_{1r} - C_{1s} - Z_{1s} > Z_{0r} - C_{0s} - Z_{0s} \quad (18)$$

$$P_{1r} > P_{0r}, \quad \text{i.e. } S_1 - C_{1r} - Z_{1r} > S_0 - C_{0r} - Z_{0r} \quad (19)$$

Based on the constraints (17), (18), and (19), an isosceles right triangle TT'T" is drawn in figure 9, which is a *profit triangle* before information sharing, because any point in the triangle represents transfer costs (Z_{0s}, Z_{0r}) such that everyone has profit. Individual profit is measured by the distance from that point to a corresponding side of the triangle (e.g., the distance from a point x^* to TT" equals the supplier's profit divided by $\sqrt{2}$). Point T denotes the transfer costs (Z_{0r}, Z_{0s}) which give the retailer all of the profit. Point T' denotes the transfer costs (Z_{0r}, Z_{0s}) which give the supplier all of the profit. Point T" denotes the transfer costs (Z_{0r}, Z_{0s}) which give the manufacturer all of the profit. Total profit is measured by the length of a shorter side in the *profit triangle*.

Usually after information sharing, $S_1 - C_{1r} > S_0 - C_{0r}$ since the retailer's sale increases, while $C_{1s} < C_{0s}$ since the supplier can order appropriately to reduce setup cost by tracking retailer's inventory (for instance, Model III in Figure 4.4). Whether $C_{1m} < C_{0m}$ or $C_{1m} > C_{0m}$ depends on the parameters of the specific problem. Figure 4.10 describes a situation where $C_{1m} < C_{0m}$. The triangle WW'W" is a *profit triangle* after information sharing, which is drawn based on the constraints (14), (15) and (16). By satisfying constraints (17), (18), and (19), an *information sharing triangle* UU'U" (also an isosceles right triangle) is drawn, which defines the area where everyone may make additional profit after information sharing (refer to figure 10). Given a specific NIS transfer cost point x^* (the transfer cost between the supplier and retailer is x_r^*, and the transfer cost between the supplier and the manufacturer is x_s^*), an IS transfer cost point y^* (the transfer cost between the supplier and the retailer is y_r^*, and the transfer cost between the supplier and the manufacturer is y_s^*) must be within UU'U" such that every

one is willing to implement information sharing. Given the *x** represented by (x_r^*, x_s^*), it is easy to figure out the vertices of the *information sharing triangle* UU'U", say

$$U: (x_r^* + C_{1m} - C_{0m} + C_{1s} - C_{0s}, x_s^* + C_{1m} - C_{0m}),$$

$$U': (x_r^* + S_1 - C_{1r} - S_0 + C_{0r}, x_s^* + C_{1m} - C_{0m}),$$

$$U'': (x_r^* + S_1 - C_{1r} - S_0 + C_{0r}, x_s^* + S_1 - C_{1r} - S_0 + C_{0r} - C_{1s} + C_{0s}).$$

The additional profit for each supply chain member, which comes from information sharing, is measured by the distance from the point *y** to a corresponding side of UU'U". It is interesting to note that no matter how the NIS transfer costs *x** moves within TT'T", the shape and size of UU'U" does not change, and UU'U" moves accordingly within WW'W".

If the cost of information sharing (CI) is considered, the *profit triangle* after information sharing and the *information sharing triangle* will shrink. Figure 4.11 gives an example of how WW'W" and UU'U" shrink if all the cost of information sharing is charged to the retailer (for instance, Model III in Figure 4.4). If the CI is too large, UU'U" may not exist. In this case, information sharing is not worth implementing.

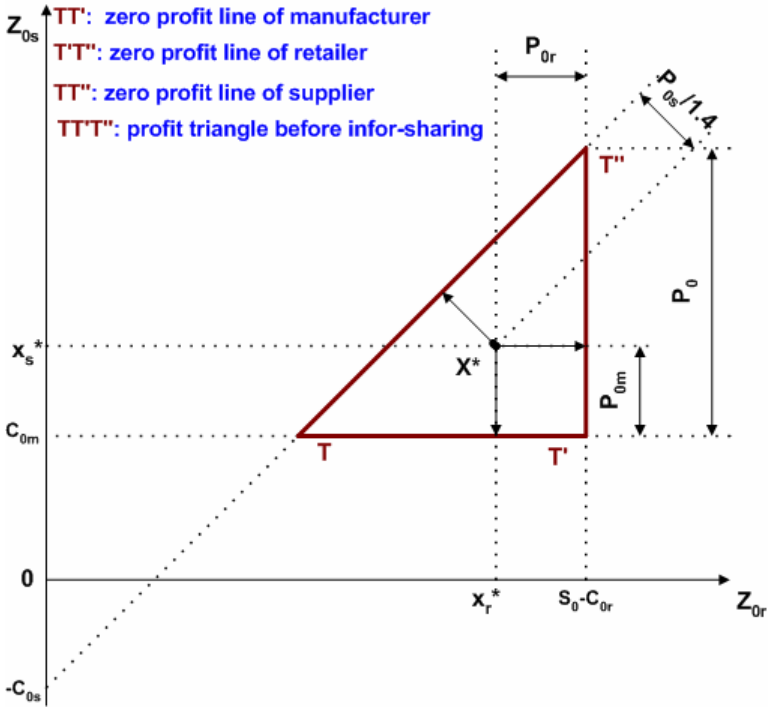

Figure 4.9: Profit Triangles before Information Sharing

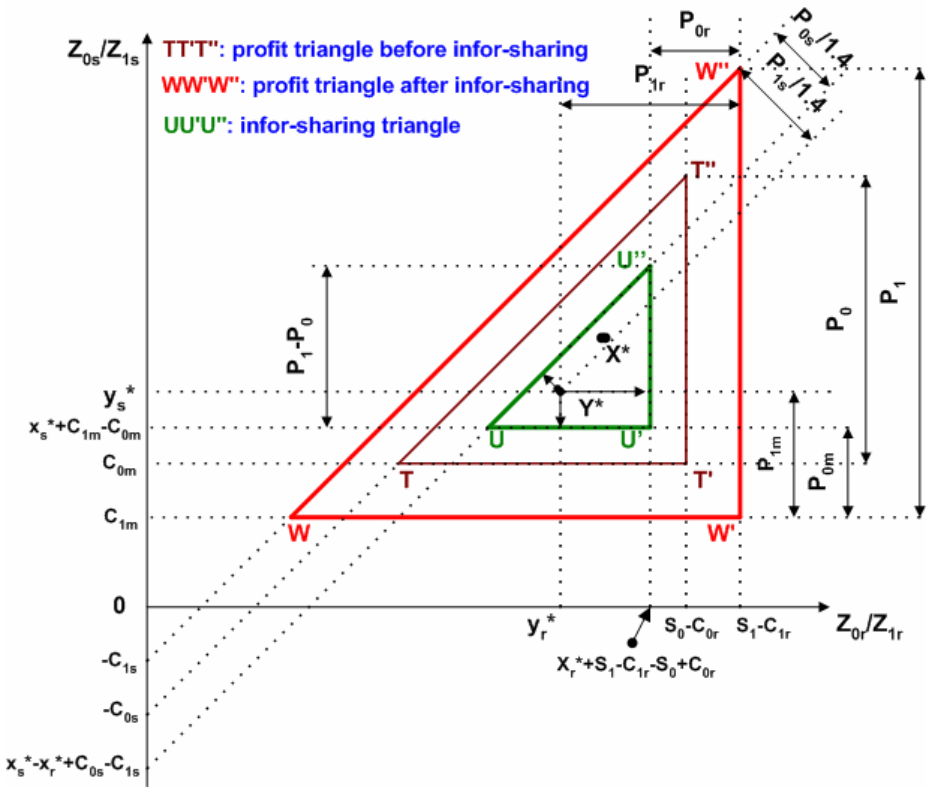

Figure 4.10: Information Sharing Triangle

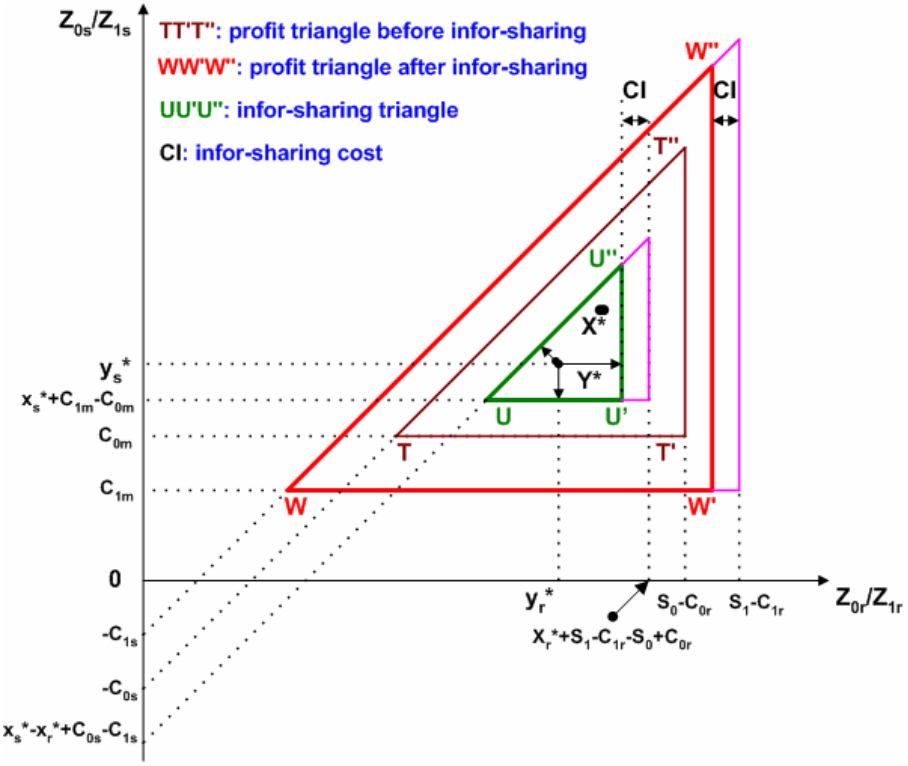

Figure 4.11: Information Sharing Triangle Shrinks When CI is charged to the Retailer

4.5 Conclusion

The MDP approach provides a powerful tool for the study of multi-stage supply chains. The *evolutionary coordination* algorithm coupled with *policy perturbations* and *multiple starts* help to solve the supply chain problem effectively and efficiently. Experimentation indicates that appropriate information sharing may bring significant benefit to the supply chain, while some information sharing may not be valuable. Depending on the cost of information sharing, it may, or may not, be worthwhile to implement information sharing.

This chapter also discusses the issue of transfer cost negotiation within a supply chain, when members of the chain may be from different cost centers or even different companies. A framework is provided within which the negotiations necessarily have to occur, thus giving guidance for determination of the transfer price or unit purchase price for each of the supply chain members. Future research will explore this issue in more detail. Other issues to be considered may include unsynchronized lead times, multiple products, and alternative supply structures.

4.6 References

Cachon, G.P. and M. Fisher, 2000, Supply chain inventory management and the value of shared information, *Management Science*, vol. 45, 843-856.

Cohen, S.L., 2000, Asymmetric information in vendor managed inventory systems, *unpublished PhD Thesis*, Stanford University.

D'Amours, S., B. Montreuil, P. Lefrancois and F. Soumis, 1999, Networked manufacturing: the impact of information sharing, *International Journal of Production Economics,* 58, 63-79.

Davis, L.B., 2004, State Clustering in Markov Decisions Processes with an Application in Information Sharing, unpublished *Ph.D. dissertation*, Industrial Engineering Department, N.C. State University.

Ding, F.Y., T.J. Hodgson and R.E. King, 1988, A methodology for computation reduction for specially structured large scale Markov decision problems, *European Journal of Operational Research,* Vol. 34, 105-112.

Gavirneni, S., R. Kapuscin and S. Tayur, 1999, Value of information in capacitated supply chains, *Management Science*, 46, no.1, 16-24.

Gavirneni, S., 2002, Information flows in capacitated supply chains with fixed ordering cost, *Management Science*, 48, 644-651.

Huang, G.Q., J.S.K. Lau, and K.L. Mak, 2003, The impacts of sharing production information on supply chain dynamics: a review of the literature, *International Journal of Production Research,* 41, no.7, 1483-1517.

Hodgson, T.J., and D. Wang, 1992, "Computation reduction for large scale MDPs," *International Journal of Production Research*, Vol. 30, No. 10.

Howard, R., 1960, *Dynamic Programming and Markov Processes,* MIT Press, Cambridge, MA

Lee, H., Padmanabhan, V. and Whang, S. 1997, Information distortion in a supply chain: the bullwhip effect, *Management Science*, 43, 546-558.

Lee, H.L., K.C. So, and C.S. Tang, 2000, The value of information sharing in a two-level supply chain, *Management Science*, 46, 626-643.

Ng, C.T., Y.O. Li, and K. Chakhlevitch, 2001, Coordinated replenishments with alternative supply sources in two-level supply chains, *Int'l Journal of Production Economics,* 73, 227-240.

Scarf, H., 1960, The Optimality of *(S,s)* Policies in the dynamic inventory problems, *Mathematical Methods in the Social Science*, K.J. Arrow, S. Karlin, and P. Suppes (Eds.), Stanford University Press.

Shaoxiang, C. and Lambrecht, M., "X-Y band and modified (s,S) policy", *Operations Research*, 44/6 (1996) 1013-1019.

Simchi-Levi, D. and Y. Zhao, 2002, The value of information sharing in a two-stage supply chain with production capacity constraints: the infinite horizon case, *Manufacturing and Service Operations Management*, 4, no.1, 21-24.

Swaminathan, J. M., N.M. Sadeh and S.F. Smith, 1997, Effect of sharing supplier capacity information, Haas School of Business, University of California - Berkeley.

Tan, G. W., 1999, The impact of demand information sharing on supply chain network, unpublished *Ph.D. dissertation*, University of Illinois at Urbana-Champaign.

Wagner, H.M., 1972, *Principles of Operations Research,* Appendix II, Prentice-Hall International Editions.

White, D.J., 1960, Dynamic programming, Markov chains, and method of successive approximations, *Journal of Mathematical Analysis and Application*, 6, 373-376.

4.7 Appendix

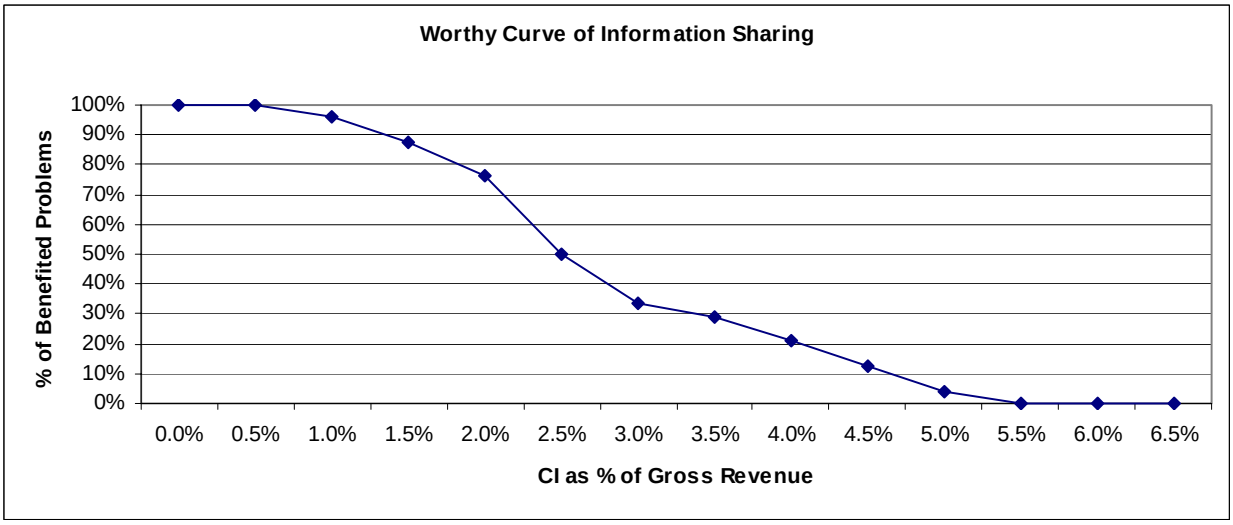

Figure 4.12: Worthy Curve of Information Sharing for 96 Problem Instances in DOE2

Table 4.6: Worthy Curve Analysis for 96 Problem Instances in DOE2 when CI=0[14]

#Prob	CI	Sales0	worthy	x	y_min	y	y_max	W0=x/q0	W1_min	W1=y/q1	W1_max
1	0	496.651	1	280.64	277.9	281.4	284.9	56.51	55.69	56.40	57.10
2	0	489.426	1	275.05	269.23	275.14	281.04	56.20	54.67	55.86	57.06
3	0	475.251	1	261.29	253.19	259.34	265.49	54.98	53.04	54.32	55.61
4	0	497.722	1	281.98	276.3	280.9	285.4	56.65	55.37	56.29	57.21
5	0	491.66	1	275.43	266.10	272.69	279.27	56.02	53.98	55.31	56.65
6	0	476.728	1	252.84	241.42	248.26	255.09	53.04	50.56	51.99	53.42
7	0	495.779	1	288.57	280.83	286.82	292.80	58.21	56.39	57.59	58.79
8	0	485.741	1	278.63	272.65	281.90	291.15	57.36	55.40	57.28	59.16
9	0	471.903	1	266.11	254.94	265.20	275.47	56.39	53.48	55.64	57.79
10	0	496.654	1	288.91	280.70	288.07	295.44	58.17	56.26	57.74	59.22
11	0	488.574	1	278.89	267.24	278.86	290.48	57.08	54.26	56.62	58.98
12	0	474.85	1	260.23	243.19	255.48	267.77	54.80	50.94	53.51	56.09
13	0	496.651	1	305.47	302.9	306.3	309.7	61.51	60.72	61.40	62.08
14	0	489.426	1	299.52	294.01	299.76	305.51	61.20	59.70	60.86	62.03
15	0	474.85	1	284.26	277.17	283.21	289.25	59.86	58.06	59.32	60.59
16	0	497.722	1	306.87	301.3	305.8	310.3	61.65	60.39	61.29	62.20

[14] This table analyzes the worthy curve for a two-stage supply chain. The information sharing case is based on Model III and no information sharing is based on Model IV. "CI" denotes the cost of information sharing. "Sales0" denotes the gross revenue when there is no information sharing. "x" denotes the transfer cost for no information sharing case such that the profit is allocated among retailer and supplier evenly. "y" denotes the transfer cost for information sharing case such that the profit is allocated evenly. "y_min" and "y_max" are the minimal and maximal transfer costs that both retailer and supplier will be benefited from information sharing, respectively. "q0" and "q1" are the average transfer quantities per period that keep profit allocated evenly before and after information sharing, respectively. "W0" and "W1" denote the unit purchase prices that keep profit allocated evenly before and after information sharing, respectively. "W1_min" and "W1_max" are the limits for the unit purchase price after information sharing if no information sharing transfer cost is "x". If "worthy" value equals 1, the implementation of information sharing is worthy. Since CI=0, all the problems will be benefited from information sharing.

Table 4.6 (Continued)

17	0	491.66	1	300.01	290.58	297.10	303.62	61.02	58.95	60.28	61.60
18	0	476.728	1	276.68	265.33	272.13	278.93	58.04	55.57	56.99	58.42
19	0	495.779	1	313.36	305.84	311.72	317.59	63.21	61.41	62.59	63.77
20	0	485.741	1	302.92	296.11	305.07	314.03	62.36	60.25	62.08	63.90
21	0	471.903	1	289.70	278.86	288.89	298.91	61.39	58.51	60.61	62.72
22	0	496.654	1	313.74	305.76	313.01	320.27	63.17	61.28	62.74	64.19
23	0	488.574	1	303.32	292.06	303.49	314.91	62.08	59.30	61.62	63.94
24	0	474.85	1	283.97	267.19	279.35	291.51	59.80	55.97	58.51	61.06
25	0	496.65	1	275.84	273.1	276.5	279.9	55.54	54.74	55.42	56.09
26	0	489.426	1	270.16	264.34	270.15	275.96	55.20	53.67	54.85	56.03
27	0	475.251	1	256.26	248.17	254.27	260.37	53.92	51.98	53.26	54.54
28	0	497.721	1	277.12	271.4	275.9	280.4	55.68	54.40	55.30	56.21
29	0	491.66	1	270.46	260.91	267.45	273.99	55.01	52.93	54.26	55.59
30	0	476.728	1	247.78	236.36	243.19	250.01	51.98	49.50	50.93	52.36
31	0	495.779	1	283.81	276.09	281.97	287.85	57.25	55.44	56.62	57.80
32	0	485.741	1	273.82	266.44	275.55	284.65	56.37	54.22	56.07	57.92
33	0	471.903	1	261.16	249.88	260.05	270.22	55.34	52.43	54.56	56.69
34	0	496.654	1	284.11	275.91	283.15	290.38	57.21	55.30	56.75	58.20
35	0	488.574	1	274.01	262.36	273.87	285.38	56.08	53.27	55.61	57.94
36	0	474.85	1	255.21	238.18	250.41	262.64	53.75	49.89	52.45	55.02
37	0	496.641	1	300.68	298.2	301.4	304.7	60.54	59.76	60.42	61.07
38	0	489.426	1	294.63	289.12	294.77	300.43	60.20	58.70	59.85	61.00
39	0	474.85	1	279.25	272.11	278.10	284.08	58.81	57.00	58.25	59.51
40	0	496.651	1	299.00	296.5	300.9	305.3	60.20	59.42	60.30	61.19
41	0	491.66	1	295.05	285.30	291.78	298.26	60.01	57.90	59.21	60.53
42	0	476.728	1	271.62	260.26	267.05	273.83	56.98	54.51	55.93	57.35
43	0	495.779	1	308.60	301.13	306.90	312.67	62.25	60.47	61.63	62.79
44	0	485.741	1	298.11	291.30	300.12	308.94	61.37	59.27	61.07	62.86
45	0	471.903	1	284.75	273.91	283.84	293.78	60.34	57.47	59.56	61.64
46	0	496.654	1	308.95	300.97	308.09	315.22	62.21	60.32	61.75	63.18
47	0	488.574	1	298.44	287.18	298.50	309.81	61.08	58.31	60.61	62.90
48	0	474.85	1	278.96	262.17	274.28	286.39	58.75	54.92	57.45	59.99
49	0	496.631	1	284.61	282.5	284.9	287.2	57.31	56.73	57.20	57.67
50	0	489.426	1	279.38	275.8	280.6	285.3	57.08	56.01	56.97	57.93
51	0	475.251	1	265.95	258.47	263.61	268.75	55.96	54.22	55.30	56.38
52	0	496.651	1	282.97	282.6	286.1	289.5	56.97	56.65	57.34	58.03
53	0	491.66	1	280.01	271.32	276.92	282.53	56.95	55.09	56.23	57.36
54	0	476.728	1	257.69	247.93	253.89	259.84	54.05	51.93	53.18	54.42
55	0	495.779	1	292.51	286.8	291.8	296.7	59.00	57.59	58.59	59.59
56	0	485.741	1	282.77	277.60	285.71	293.83	58.21	56.49	58.14	59.79
57	0	471.903	1	270.56	261.34	270.55	279.77	57.33	54.83	56.77	58.70
58	0	496.651	1	292.91	287.07	293.22	299.37	58.98	57.54	58.77	60.01
59	0	488.574	1	283.20	273.82	284.24	294.66	57.96	55.60	57.72	59.83
60	0	474.85	1	264.88	249.21	260.47	271.74	55.78	52.23	54.59	56.95
61	0	496.631	1	309.44	307.5	309.8	312.1	62.31	61.75	62.20	62.66
62	0	489.426	1	303.86	300.4	305	309.5	62.08	61.01	61.94	62.87
63	0	474.85	1	288.91	282.35	287.41	292.46	60.84	59.24	60.30	61.36

Table 4.6 (Continued)

64	0	496.651	1	307.80	307.7	311	314.3	61.97	61.67	62.34	63.00
65	0	491.66	1	304.59	295.98	301.54	307.11	61.95	60.10	61.23	62.35
66	0	476.728	1	281.52	271.84	277.76	283.68	59.05	56.94	58.18	59.41
67	0	495.779	1	317.30	311.8	316.7	321.5	64.00	62.61	63.59	64.56
68	0	485.741	1	307.06	302.46	310.29	318.11	63.21	61.55	63.14	64.73
69	0	468.436	1	287.34	285.15	294.13	303.11	61.34	59.85	61.73	63.62
70	0	495.779	1	315.19	312.13	318.16	324.19	63.57	62.57	63.77	64.98
71	0	488.574	1	307.63	298.64	308.86	319.09	62.96	60.64	62.72	64.79
72	0	474.85	1	288.62	272.80	283.95	295.10	60.78	57.19	59.52	61.86
73	0	496.617	1	279.82	277.7	280	282.3	56.34	55.77	56.22	56.68
74	0	489.426	1	274.49	270.7	275.4	280	56.08	54.99	55.93	56.87
75	0	474.85	1	260.16	253.41	258.53	263.64	54.79	53.16	54.24	55.31
76	0	496.651	1	278.17	277.9	281.1	284.4	56.01	55.69	56.35	57.01
77	0	491.66	1	275.04	266.35	271.93	277.51	55.94	54.08	55.21	56.35
78	0	476.728	1	252.63	242.87	248.81	254.76	52.99	50.87	52.11	53.36
79	0	495.779	1	287.75	282.1	286.9	291.8	58.04	56.64	57.61	58.58
80	0	485.741	1	277.96	272.79	280.76	288.73	57.22	55.51	57.13	58.76
81	0	471.903	1	265.61	256.14	265.27	274.39	56.28	53.76	55.67	57.59
82	0	495.782	1	285.65	282.27	288.29	294.31	57.62	56.58	57.79	58.99
83	0	488.574	1	278.32	268.94	279.25	289.56	56.97	54.61	56.70	58.80
84	0	474.85	1	259.87	243.83	255.04	266.26	54.73	51.11	53.46	55.81
85	0	496.61	1	304.64	302.7	304.9	307.1	61.34	60.78	61.22	61.67
86	0	489.426	1	298.96	295.5	300	304.5	61.08	60.02	60.93	61.84
87	0	474.85	1	283.90	277.33	282.36	287.39	59.79	58.18	59.24	60.29
88	0	496.651	1	303.00	302.9	306.1	309.3	61.01	60.71	61.35	61.99
89	0	491.66	1	299.63	291.02	296.55	302.09	60.94	59.09	60.21	61.34
90	0	476.728	1	276.46	266.78	272.69	278.60	57.99	55.88	57.11	58.35
91	0	495.779	1	312.54	307.1	311.8	316.5	63.04	61.66	62.61	63.56
92	0	485.741	1	302.25	297.65	305.33	313.02	62.22	60.57	62.13	63.70
93	0	468.436	1	282.45	280.26	289.09	297.92	60.30	58.82	60.67	62.53
94	0	495.782	1	310.44	307.03	312.90	318.76	62.62	61.56	62.73	63.91
95	0	488.574	1	302.75	293.76	303.88	313.99	61.97	59.65	61.70	63.76
96	0	474.85	1	283.61	266.74	277.85	288.97	59.73	55.96	58.29	60.62

Table 4.7: Worthy Curve Analysis for 96 Problem Instances in DOE2 when CI=2%*Sales0[15]

#Prob	CI	Sales0	worthy	*x*	y_min	*y*	y_max	W4=x/q0	W3_min	W3=y/q1	W3_max
1	9.93	496.651	0	280.64	-1	-1	-1	56.51	-1	-1	-1
2	9.79	489.426	1	275.05	279.02	280.03	281.04	56.20	56.65	56.86	57.06
3	9.51	475.251	1	261.29	262.70	264.09	265.49	54.98	55.03	55.32	55.61
4	9.95	497.722	0	281.98	-1	-1	-1	56.65	-1	-1	-1
5	9.83	491.66	1	275.43	275.94	277.60	279.27	56.02	55.97	56.31	56.65

[15] If "worthy" value equals 0, the cost of information sharing is too great, i.e., the information sharing for that supply chain problem is not worthy, and corresponding values are filled as -1 for "y_min", "y", "y_max", "W3_min", "W3" and "W3_max."

Table 4.7 (Continued)

6	9.53	476.728	1	252.84	250.96	253.02	255.09	53.04	52.56	52.99	53.42
7	9.92	495.779	1	288.57	290.74	291.77	292.80	58.21	58.38	58.59	58.79
8	9.71	485.741	1	278.63	282.36	286.76	291.15	57.36	57.37	58.26	59.16
9	9.44	471.903	1	266.11	264.38	269.92	275.47	56.39	55.46	56.63	57.79
10	9.93	496.654	1	288.91	290.63	293.03	295.44	58.17	58.25	58.73	59.22
11	9.77	488.574	1	278.89	277.01	283.75	290.48	57.08	56.24	57.61	58.98
12	9.5	474.85	1	260.23	252.69	260.23	267.77	54.80	52.93	54.51	56.09
13	9.93	496.651	0	305.47	-1	-1	-1	61.51	-1	-1	-1
14	9.79	489.426	1	299.52	303.80	304.66	305.51	61.20	61.68	61.86	62.03
15	9.5	474.85	1	284.26	286.67	287.96	289.25	59.86	60.05	60.32	60.59
16	9.95	497.722	0	306.87	-1	-1	-1	61.65	-1	-1	-1
17	9.83	491.66	1	300.01	300.41	302.01	303.62	61.02	60.95	61.27	61.60
18	9.53	476.728	1	276.68	274.87	276.90	278.93	58.04	57.57	57.99	58.42
19	9.92	495.779	1	313.36	315.76	316.67	317.59	63.21	63.40	63.59	63.77
20	9.71	485.741	1	302.92	305.82	309.93	314.03	62.36	62.23	63.07	63.90
21	9.44	471.903	1	289.70	288.29	293.60	298.91	61.39	60.49	61.60	62.72
22	9.93	496.654	1	313.74	315.69	317.98	320.27	63.17	63.27	63.73	64.19
23	9.77	488.574	1	303.32	301.83	308.37	314.91	62.08	61.28	62.61	63.94
24	9.5	474.85	1	283.97	276.68	284.10	291.51	59.80	57.96	59.51	61.06
25	9.93	496.65	0	275.84	-1	-1	-1	55.54	-1	-1	-1
26	9.79	489.426	1	270.16	274.13	275.04	275.96	55.20	55.66	55.84	56.03
27	9.51	475.251	1	256.26	257.68	259.03	260.37	53.92	53.98	54.26	54.54
28	9.95	497.721	0	277.12	-1	-1	-1	55.68	-1	-1	-1
29	9.83	491.66	1	270.46	270.74	272.36	273.99	55.01	54.93	55.26	55.59
30	9.53	476.728	1	247.78	245.90	247.95	250.01	51.98	51.50	51.93	52.36
31	9.92	495.779	1	283.81	286.00	286.93	287.85	57.25	57.43	57.62	57.80
32	9.71	485.741	1	273.82	276.16	280.41	284.65	56.37	56.19	57.06	57.92
33	9.44	471.903	1	261.16	259.31	264.77	270.22	55.34	54.41	55.55	56.69
34	9.93	496.654	1	284.11	285.84	288.11	290.38	57.21	57.29	57.75	58.20
35	9.77	488.574	1	274.01	272.13	278.76	285.38	56.08	55.25	56.60	57.94
36	9.5	474.85	1	255.21	247.67	255.16	262.64	53.75	51.88	53.45	55.02
37	9.93	496.641	0	300.68	-1	-1	-1	60.54	-1	-1	-1
38	9.79	489.426	1	294.63	298.91	299.67	300.43	60.20	60.69	60.84	61.00
39	9.5	474.85	1	279.25	281.61	282.84	284.08	58.81	58.99	59.25	59.51
40	9.93	496.651	0	299.00	-1	-1	-1	60.20	-1	-1	-1
41	9.83	491.66	1	295.05	295.13	296.70	298.26	60.01	59.89	60.21	60.53
42	9.53	476.728	1	271.62	269.79	271.81	273.83	56.98	56.50	56.93	57.35
43	9.92	495.779	1	308.60	311.05	311.86	312.67	62.25	62.46	62.62	62.79
44	9.71	485.741	1	298.11	301.01	304.97	308.94	61.37	61.25	62.06	62.86
45	9.44	471.903	1	284.75	283.34	288.56	293.78	60.34	59.45	60.55	61.64
46	9.93	496.654	1	308.95	310.90	313.06	315.22	62.21	62.31	62.75	63.18
47	9.77	488.574	1	298.44	296.96	303.38	309.81	61.08	60.29	61.60	62.90
48	9.5	474.85	1	278.96	271.67	279.03	286.39	58.75	56.91	58.45	59.99
49	9.93	496.631	0	284.61	-1	-1	-1	57.31	-1	-1	-1
50	9.79	489.426	0	279.38	-1	-1	-1	57.08	-1	-1	-1
51	9.51	475.251	1	265.95	267.98	268.36	268.75	55.96	56.22	56.30	56.38
52	9.93	496.651	0	282.97	-1	-1	-1	56.97	-1	-1	-1

Table 4.7 (Continued)

53	9.83	491.66	1	280.01	281.15	281.84	282.53	56.95	57.08	57.22	57.36
54	9.53	476.728	1	257.69	257.47	258.65	259.84	54.05	53.93	54.17	54.42
55	9.92	495.779	1	292.51	296.7	296.7	296.7	59.00	59.58	59.58	59.59
56	9.71	485.741	1	282.77	287.32	290.57	293.83	58.21	58.47	59.13	59.79
57	9.44	471.903	1	270.56	270.78	275.27	279.77	57.33	56.81	57.76	58.70
58	9.93	496.651	1	292.91	297.00	298.19	299.37	58.98	59.53	59.77	60.01
59	9.77	488.574	1	283.20	283.59	289.13	294.66	57.96	57.59	58.71	59.83
60	9.5	474.85	1	264.88	258.71	265.22	271.74	55.78	54.22	55.58	56.95
61	9.93	496.631	0	309.44	-1	-1	-1	62.31	-1	-1	-1
62	9.79	489.426	0	303.86	-1	-1	-1	62.08	-1	-1	-1
63	9.5	474.85	1	288.91	291.85	292.15	292.46	60.84	61.23	61.29	61.36
64	9.93	496.651	0	307.80	-1	-1	-1	61.97	-1	-1	-1
65	9.83	491.66	1	304.59	305.81	306.46	307.11	61.95	62.09	62.22	62.35
66	9.53	476.728	1	281.52	281.37	282.52	283.68	59.05	58.93	59.17	59.41
67	9.92	495.779	0	317.30	-1	-1	-1	64.00	-1	-1	-1
68	9.71	485.741	1	307.06	312.17	315.14	318.11	63.21	63.53	64.13	64.73
69	9.37	468.436	1	287.34	294.52	298.81	303.11	61.34	61.81	62.71	63.62
70	9.92	495.779	1	315.19	322.05	323.12	324.19	63.57	64.55	64.77	64.98
71	9.77	488.574	1	307.63	308.41	313.75	319.09	62.96	62.63	63.71	64.79
72	9.5	474.85	1	288.62	282.30	288.70	295.10	60.78	59.18	60.52	61.86
73	9.93	496.617	0	279.82	-1	-1	-1	56.34	-1	-1	-1
74	9.79	489.426	0	274.49	-1	-1	-1	56.08	-1	-1	-1
75	9.5	474.85	1	260.16	262.91	263.28	263.64	54.79	55.16	55.23	55.31
76	9.93	496.651	0	278.17	-1	-1	-1	56.01	-1	-1	-1
77	9.83	491.66	1	275.04	276.19	276.85	277.51	55.94	56.08	56.21	56.35
78	9.53	476.728	1	252.63	252.40	253.58	254.76	52.99	52.87	53.11	53.36
79	9.92	495.779	0	287.75	-1	-1	-1	58.04	-1	-1	-1
80	9.71	485.741	1	277.96	282.51	285.62	288.73	57.22	57.49	58.12	58.76
81	9.44	471.903	1	265.61	265.58	269.99	274.39	56.28	55.74	56.66	57.59
82	9.92	495.782	1	285.65	292.19	293.25	294.31	57.62	58.57	58.78	58.99
83	9.77	488.574	1	278.32	278.72	284.14	289.56	56.97	56.60	57.70	58.80
84	9.5	474.85	1	259.87	253.32	259.79	266.26	54.73	53.10	54.46	55.81
85	9.93	496.61	0	304.64	-1	-1	-1	61.34	-1	-1	-1
86	9.79	489.426	0	298.96	-1	-1	-1	61.08	-1	-1	-1
87	9.5	474.85	1	283.90	286.83	287.11	287.39	59.79	60.18	60.23	60.29
88	9.93	496.651	0	303.00	-1	-1	-1	61.01	-1	-1	-1
89	9.83	491.66	1	299.63	300.85	301.47	302.09	60.94	61.08	61.21	61.34
90	9.53	476.728	1	276.46	276.31	277.45	278.60	57.99	57.87	58.11	58.35
91	9.92	495.779	0	312.54	-1	-1	-1	63.04	-1	-1	-1
92	9.71	485.741	1	302.25	307.36	310.19	313.02	62.22	62.55	63.12	63.70
93	9.37	468.436	1	282.45	289.63	293.77	297.92	60.30	60.79	61.66	62.53
94	9.92	495.782	1	310.44	316.95	317.85	318.76	62.62	63.55	63.73	63.91
95	9.77	488.574	1	302.75	303.53	308.76	313.99	61.97	61.64	62.70	63.76
96	9.5	474.85	1	283.61	276.23	282.60	288.97	59.73	57.95	59.29	60.62

Chapter 5 Summary and Future Research

5.1 Summary and Future Research

In this dissertation an *evolutionary coordination* algorithm for DEC-ROMDP problems was introduced. Coupled with a *policy perturbation* and *multiple starts* strategy, this algorithm works effectively for supply chain DEC-ROMDP problems. Any improvement in the algorithm, like the *perturbation* method, that helps to improve solving generic DEC-ROMDP problems has the potential to make this algorithm widely used. *Successive approximation* (White 1960) is used in the algorithm to reduce computations. In order to solve extremely large-scale supply chain problems more efficiently, the *encapsulation evolutionary* approach (Ding *et al*. 1988) should be pursued in future research. Note that the *encapsulation evolution* approach has two assumptions: first, the chain structure of the MDP problem is uni-chain; and second, all possible policies are aperiodic. Unfortunately, the supply chain ROMDP or DEC-ROMDP problems may be multi-chain structured, and some implementable policies may be periodic. Davis (2004) applied the *encapsulation evolutionary* approach to the supply chain ROMDP problem, and found computational effort to be greatly reduced. Insight from Davis' work should be brought into the application of the *encapsulation evolutionary* approach to the supply chain DEC-ROMDP problems.

This dissertation quantifies the value of information sharing in a serial supply chain (both two-stage and three-stage). Future research may explore information sharing in a divergent, convergent, or network supply chain. Other issues to be considered may include unsynchronized lead times, multiple products, and backlog. The incorporation of the above issues may significantly increase the number of system states. For instance, only a single product in the supply chain is considered here. If two different products are considered, the system state space will expand greatly. If unsynchronized lead times or backlog comes into play, the system state representation becomes complicated since it is related to not only inventory levels, but also lead time or back orders. However, the evolutionary coordination algorithm can still be applied to these more complicated situations.

The issue of transfer cost negotiation within a supply chain is discussed, when members of the chain may be from different cost centers or even different companies. This dissertation provides a framework within which the negotiations necessarily have to occur, thus giving guidance for determination of the transfer price for supply chain members. Future research should explore this issue in more detail.

5.2 References

Davis, L.B., 2004, State Clustering in Markov Decisions Processes with an Application in Information Sharing, unpublished *Ph.D. dissertation*, Industrial Engineering Department, N.C. State University.

Ding, F.Y., T. Hodgson and R. King, 1988. A methodology for computation reduction for specially structured large scale Markov decision problems. *European Journal of Operational Research*. *Volume* 34:105-112.

White, D.J. 1960. Dynamic programming, Markov chains, and method of successive approximations. *Journal of Mathematical Analysis and Application*, 6, 373-376.