APPROVAL SHEET

Title of Dissertation: An Extended Bayesian Belief Network Model of Multi-agent Systems for Supply Chain Management

Name of Candidate: Ye Chen

Doctor of Philosophy, 2001

Dissertation and Abstract Approved: ____________________________

____

____

____

Date Approved: ____________________

Curriculum Vita

Name: Ye Chen.

Permanent address: 841 Bayridge Drive
Gaithersburg, MD, 20878.

Degree and date to be conferred: Ph.D., 2001.

Date of Birth: April 21, 1973.

Place of Birth: Wuhan, Hubei province, People's Republic of China.

Secondary education:

June 1991 The affiliated high school of South China Normal University.

Collegiate institutions attended:

May 2001	Ph.D.,	University of Maryland, Baltimore County.
May 1998	M.S.,	University of Maryland, Baltimore County.
June 1995	B.A.,	South China University of Technology.

Major: Computer Science.

Professional publications:

Ye Chen et al., “A Negotiation-based Multi-agent System for Supply Chain Management,” in *work note of Autonomous Agents’99 workshop on Agent-based Decision-support for Managing the Internet-enabled supply-chain*, Seattle, Washington State, U.S, May 1, 1999.

R. Scott Cost, Ye Chen, Tim Finin, Yannis Labrou, Yun Peng, “Modeling Agent Conversations with Colored Petri Nets.” In *Working Notes of the Workshop on Specifying and Implementing Conversation Policies, Third International Conference on Autonomous Agents (Agents '99)*, Seattle, Washington, U.S., May 1, 1999.

R. Scott Cost, Ye Chen, Tim Finin, Yannis Labrou, Yun Peng, “Using Colored Petri Nets for Conversation Modeling.” in *Working Notes of the Workshop on Agent Communication Languages, Sixteenth International Joint Conference on Artificial Intelligence (IJCAI '99),* Stockholm, Sweden, August 1, 1999.

Ye Chen et al., *“Negotiating agents for Supply Chain Management,” in work note of AAAI workshop of electronic commerce,* Orlando Florida, U.S., July 31 - August 4, 1999.

Abstract

Title of dissertation: An Extended Bayesian Belief Network Model of

Multi-agent Systems for Supply Chain Management

Ye Chen, Doctor of Philosophy, 2001

Dissertation Directed by: Yun Peng, Associate Professor,

Department of Computer Science and Electronic Engineering

This dissertation develops a theoretical model, called an extended Bayesian Belief Network (eBBN), of a Multi-agent System for Supply Chain Management (MASCM), which formalizes agent interactions in uncertain environments.

MASCM is an electronic marketplace as well as a supply chain management system where agents sell and buy products on behalf of their owners to gain profits. A virtual chain consists of agents connected by commitments triggered by an end order. The system performance is measured by whether the management goal, e.g. end customer satisfaction, shared by all virtual chains can be reached.

Due to the uncertain nature of internal and external decision factors, a commitment made by an agent may eventually not be fulfilled. Uncertainty concerning one agent's commitments may propagate over the chain via its supplier-customer

connections. Uncertainty and its propagation may have negative impacts on agents' operations, cause inventories to be increased, the chain to be disturbed or destroyed, and eventually end orders to be delayed.

To reduce potential damage from uncertainty, agents may choose to cooperate with each other by sharing information. This type of agent interaction in uncertain environments is formalized as eBBN, in which the effects of uncertainty are modeled as agents' beliefs about the failure of commitments, relationships between these beliefs as direct causal links, and information sharing as belief update and propagation. By properly incorporating actions and their consequences into the network, eBBN further extends the representation and inference capability of traditional Bayesian Belief Networks (BBNs). The model can not only reason about the effects of agents' strategic behaviors in updating beliefs but can also describe dynamic causal structures as virtual chains evolve over time.

As a formal model, eBBN provides a sound basis for developing effective algorithms of uncertainty management. It can serve as an analytic platform to quantitatively study the relationship between agents' local behaviors and overall system performance in an uncertain environment. Several algorithms for both local decisions and global optimization have been developed and tested. The simulation results present that the system with agents using these algorithms can achieve stable performance even when uncertain events occur with high frequency.

An Extended Bayesian Belief Network Model of Multi-agent Systems for Supply Chain Management

by
Ye Chen

Dissertation submitted to the Faculty of the Graduate School
of the University of Maryland in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2001

To my parents and sisters

Acknowledgement

I am thankful to Dr. Yung Peng, Tim Finin, Chung-sheng Li, Kostas Kalpaskis, Rakesh Mohan, Tulay Adali, Reed Letsinger, and Scott Cost for serving on my committee. I am especially grateful to my advisor Dr. Yun Peng for his ideas, advice, and support without which this dissertation would not have been possible. I deeply thank Dr. Tim Finin for his wise instructions and considerable care to a student with a different cultural background.

I would like to express my special thanks to Martin Griss and Rakesh Mohan for giving me opportunities to meet many outstanding researchers at Hewlett-Parkard Laboratories and IBM T.J. Watson Research Center.

Thanks to Joy, Rong, Xiaocheng, Yongkai, Mang, Jiangyang, Linlan, Bo, Youyong, Shuang, Ting, and Lynn for their friendship.

I could never thank my parents enough for their love, guidance and support at every stage of my life. My three wonderful sisters, Cindy, Lisa and Sharon, have always given me their unbounded love. They have been my constant source of inspiration.

Last but not least I want to thank all people who have ever helped and loved me.

Table of contents

Table of figures

Table of tables

Chapter 1 Introduction

Businesses-to-businesses electronic marketplaces attract more attention as Internet technologies become increasingly available to all enterprises. Electronic supply chain management systems are special marketplaces that manage business entity networks connected by demand-supply relationships to maximally ensure that final products can be delivered to fulfill consumer's requests. Uncertainty is the disturbance that hinders system achievement of the management goals. This dissertation studies how to formalize entity efforts in their interactions to diminish the impact of uncertainty. In this chapter, we briefly introduce research background, motivation and related works.

1.1 Research background

An enterprise (firm) in the marketplace is defined as an entity that intends to gain longer-term profits for its owners or shareholders [1]. The success of an enterprise is measured by net profit that is dependent on several factors including revenues, operational fees and outsourcing expenses. If an enterprise can maximally improve customer satisfaction, that is, provide the qualified products at the right time to the right place, it can build a positive reputation and attract customers to place orders so that its revenues can increase accordingly. To create a marginal profit, the enterprise also needs to lower the internal operation cost and search for proper suppliers that can provide the materials it needs at reasonable price and also guarantee the quality of the service. This leads to the concept of a supply chain. A supply chain consists of a focal enterprise and its direct suppliers and customers. The management work of the chain is to allow the enterprise to look for an optimal solution that balances customer requests, internal operation expenses and suppliers' performance.

Recent research in supply chains indicates that, besides the direct suppliers, the indirect suppliers also affect chain management. The reasons are, firstly, behaviors of indirect suppliers can influence direct suppliers' actions, and eventually affect the focal enterprise; secondly, by knowing the situation of the suppliers' suppliers, the enterprise can gain a more accurate understanding of direct suppliers' current performance so that it can design a plausible plan for incoming customer orders. Certain research results related to the effects of indirect supplier on a focal enterprise can be found in the study of Bull-Whip phenomenon of supply chain [2]. On the other side, the behaviors of final product consumers that buy goods from retailers, called end customer or ultimate customer, reflect the trend of market change. The purchase activities might impact the enterprise decisions even for the manufacturers that provide raw materials. This is true especially when the enterprise design a long-term plan. Therefore, a modern supply chain is usually defined as an inter-organization system that can contain multiple levels of enterprises and entities, from raw material providers to end customers. The definition of a supply chain used in this dissertation is as follows,

> *A supply chain is comprised of multiple good (service) suppliers and consumers that transform raw materials into a certain product that can be delivered to the end customers.*

Similar definitions can also be found in [3]. In a supply chain each enterprise can provide a certain product (or service; in the context of this dissertation these two terms are used interchangeably without further explanations) to the others through consuming various materials. System components of a chain are organized by their market-oriented attributes. In other words, the major connection between two enterprises, if it exists, is a supply-demand relationship. The difference between a general marketplace and a supply chain system is that a supply chain only provides one type of final product to the end customers and all the entities in the system are either direct or indirect suppliers that produce parts or materials of the final product. Even though different enterprises at different positions in the chain system might have totally different views of the chain management goals, they all realize that improving end customer satisfaction will benefit all of them. Thus, from a system perspective, we define supply chain management as follows,

> *Supply chain management consists of all activities that help an enterprise to search for a solution, which can balance its current target of gains profit of maximizing the short-term profits and long term needs to attract as many end customers as possible.*

The connection between two enterprises can be set up, maintained through many ways including face to face negotiation, phone conversation, document, credit and money transfer, product delivery and so on. Usually in a particular instance of a connection, there are three types of flows coexisting as the content of the connection. They are information flow, material flow and money flow [4]. Face-to-face negotiation and phone conversation belong to information flow. Product delivery is an example of material flow content. The money or credit transfer process is the example of the money flow. Given a material or money flow, a corresponding information flow is created or deleted. For example, each product delivery process is usually accompanied by the procedure of exchanging documents with required signatures from the supplier and the customer. When the delivery truck leaves the building of a service provider, an information flow is created and started. When the receiver (service consumer) signs the receipt and it comes back to the provider, the information flow is completed and deleted. Therefore, through the study of information flows, many valuable results from various interesting aspects of a supply chain can be obtained. In fact, in this dissertation we study supply chain management as a way of analyzing information flows among enterprises.

With the rapid development of computer technologies, more and more work originally consigned to human workers is delegated to computer software and hardware. Human beings simply provide the action guidelines but let programs make automatic decisions on their behalf. They only authenticate sensitive or critical decisions. These technologies save the enterprise internal operation expenses by cutting labor costs and improve efficiency through the information circulation time reduction. As the Internet has gradually become an electronic economic platform as well as information sharing mechanism, inter-organization interactions have changed accordingly. Nowadays, Electronic Data Interchange (EDI) and eXtensible Markup Language (XML) are popular content formats for communication between enterprises. The information flows in a chain usually are bit streams consisting of 0s and 1s transmitted over TCP/IP and HTTP protocols rather than voice phone calls or meetings. Today, electronic commerce, including both business-to-business and business-to-customer types, has become a major field of economic growth. It can be predicted that in the future, the enterprise will become virtual, in other words, dependent on more automatic decision processes and digital communication and less on human interventions. From this viewpoint, as an independent business functional entity, an enterprise can be studied and analyzed as a software agent that is defined as an “intelligent, cooperative and autonomous” unit [5]. Accordingly, agent interactive actions define information flows. In this way, traditional supply chains are transformed into Multi-agent Systems (MAS). A MAS, consisting of more than one chain of virtual enterprises or software agents that are connected through demand-supply relationships and can deliver certain final good to the end customer, is one type of electronic supply chain management system and is called Multi-agent System for Supply Chain and its Management (MASCM) [6]. Like an ordinary marketplace, MASCM has to handle uncertainty since any unexpected events can damage agents’ chain management goal. For example, a strike happening at a raw material provider can eventually cause retailers to postpone the final product delivery to end customers. Moreover, with low operational cost and convenient communication channels, agents can more easily switch to other trading partners if they think changes can create more net profits. This type of action lets MASCMs become more dynamic. This dissertation research studies agent interactive behaviors and their impact on MASCM performance when the system is in an uncertain and dynamic environment.

1.2 Research motivations

Agents in the MASCM can be self-interested. They try to gain profits for their owners. However, since the end customer satisfaction improvement can bring all of them benefits, agents would like to cooperate with each other if this type of action does not harm their essential interests. Uncertain events can cause an order from an end customer to be delayed or to fail. To protect their long-term benefits, one type of cooperation that agents are willing to perform is to share information related to uncertainty. This brings out a critical problem for system study, that is, to find out an efficient way that agents can interact with each other to share uncertainty information without obligations to expose sensitive data. This leads to our research on formalizing agent interactions under uncertain and dynamic settings. In other words, the focus of our investigation is to develop a formal model that can deal with uncertainty propagation and analysis for MASCM.

As an electronic supply management system, we observed the following to be true. First, in the MASCM, the most substantial business connection between two agents is the supply-demand relationship. In this connection, the demand side's satisfaction depends on the supply side's performance. In other words, demand side and supply side are causally linked. Second, the most important task in an agent lifetime is to finish its commitments to customers and earn profits. That is, the impact of uncertain events can be quantitatively measured by their effect on the completion of commitments. Naturally, an agent's uncertainty propagation can be conducted through a casual network consisting of commitments linked by supply-demand relationships. Based on this observation, in this dissertation we present extended Bayesian Belief Network (eBBN) as the model to formalize agent interactions with uncertainty. It can be further used as an analytic platform to study the relationship between an uncertainty management mechanism and system performance.

1.3 Related works

Reflecting an increased interest in software agents, a great number of publications have appeared in academic journals, and conferences and workshops on various topics related to agent technology, including the theory, methodology, and applications of agents and multi-agent systems. Recently, researchers have started to shift their attention to theoretical modeling of multi-agent systems. The majority of work in this direction attempted to apply various economics principles (e.g., game theory, market mechanism, price system, auction, tax mechanism, etc.) to Multi-agent System (MAS) of large scales [57][59][61][62] Their rationale is that, like an economy of many active entities that interact with each other, large scale MAS can be modeled and analyzed at a macro level. Unified Modeling Language (UML) is the industry standard to design large-scale object-oriented systems. Some researchers have made an effort to extend UML and define Agent-oriented UML (AUML) that supports the agent system development cycle from analysis to final implementation maintenance [7][8][9]. Color Petri net is another potential modeling tool for agent system analysis because of its wonderful expressive power for concurrent events [10]. Some other works [11][12] use workflows to study autonomous entity interactions. The model tools have several common weak points. First, they are too general. These models are not specialized for supply chain management systems and cannot describe dynamic business flows, e.g. agent negotiation processes.

Second, they implicitly or explicitly assume that the system is designed and implemented as a whole. This is not true for a MASCM, in which system components might be designed, implemented and owned by different business entities. Lastly but most importantly, these models do not model the uncertainty in the level of detail as our model is intended to do, and they do not provide operational guidance for individual agents working in the uncertainty environment.

One line of interesting research to study uncertainty in a MAS has been reported by Penock and Wellman where they apply a security market model to some Bayesian inferences by providing a mapping from a BBN to a market system [61][62]. However, buyer and seller agents in their model are extremely simple, each corresponding to a variable in an ordinary BBN. Work in [63] proposes a framework for Probabilistic Agent Programming. The work is concentrated on individual agents, providing syntax and semantics for representing uncertain inputs an agent receives in incoming messages and a set of rules to manipulate them. However, it gives little treatment of agent interactions at the system level beyond simple message passing. Moreover, the manipulation of uncertainty with each agent is done following a set of heuristic rules for synthesizing range probabilities.

Inspired by game theory, BBN, influence diagrams, and logic programming, Poole proposes the Independence Choice Logic (ICL) to model multiple agents under uncertainty [64]. In essence, ICL extends logic and represents actions as choices from pre-defined alternatives. Using logic programming, an extended influence diagram is then formed that seeks optimal results in game theoretic terms (e.g., Nash equilibrium and Pareto optimization). We take a different approach by directly extending the BBN formulation to MAS. As such, many results in BBN research can be adopted relatively easily. ICL is more expressive than our proposed framework in that it can represent other aspect of agents such as policies or strategies. The price to pay for the expressiveness is its difficulties in using this model and its high computational complexity.

Research efforts have been made in the uncertainty community to extend the classic BBN formulation to higher order structures. The most notable work is that of the "Multiply Sectioned Belief Networks" [65]. The primary goal of their work is to improve run-time computational efficiency with BBN. This is achieved by partitioning a given ordinary BBN according to the natural division of the domain into subdomains, and limiting the computation to within one subnet most of the time. Although the objectives are different, the results from their work such as the principles guiding the formation of linkages between aggregates, the formulation of d-sepsets between subnets, and the preservation of the original joint distribution by the sectioned network are relevant to our proposed work. "Object Oriented Belief Networks" [66] attempts to extend BBN to the OO programming environment so that a BBN can be built with objects, rather than variables, as the basic units. The emphasis is on how to represent the uncertainty relationships between an object and its attributes, and how to connect objects to form a high level BBN. In contrast, our model is at a higher level of abstraction in that the internal structures of agents and their interactions are not restricted to an object oriented representation.

The issue of representing actions in the probabilistic setting is perhaps best addressed by Pearl *et al* [51][49]. Although developed for flat graphic representation of uncertainty and causality, some results from their work provide useful insights to the relationships between actions and probabilities, and will be valuable when considering incorporating actions in eBBN.

The rest of this dissertation is organized as follows. Chapter 2 clarifies the MASCM concepts, including its architecture, management activities and measurement of system performance. Chapter 3 introduces uncertainty and the basic of Bayesian Belief Network. Chapter 4 presents the extended Bayesian Belief Network (eBBN) model in details. Chapter 5 contains several algorithms using eBBN for agent decision-making procedures in different scenarios. Chapter 6 describes an experiment to study the impact of uncertainty on system performance with two different settings. System performance comparison and analysis is also given. Chapter 7 concludes dissertation research, discusses the limitations of the current model, as well as future extensions.

Chapter 2 Multi-agent System for Supply Chain Management

In the 1980's, manufacturers were faced with escalating demand for new products and the need to bring products to market with ever-increasing speed. To respond to these pressures, manufacturers were compelled to become more flexible and responsive in their manufacturing processes. As manufacturing capabilities improved in 1990s, managers realized that material and service inputs from suppliers had a major impact on the ability of their business to meet customer needs. As a result of these changes, businesses concluded that, in order to optimize their manufacturing processes, they needed to manage the network of all upstream firms that provide inputs (directly or indirectly), as well as the network of downstream firms responsible for delivery and after-market service of the product to the end customer. This leads to the inception of supply chain and chain management studies. As the development of information and networking technologies evolved, enterprises gradually transferred business responsibilities away from human beings to computer software and hardware. To some degree, an enterprise evolves into a complex information system or software agent programmed by humans to automatically fulfill business tasks. Correspondingly, the supply chain becomes an agent chain. A marketplace consisting of virtually existing agent chains that provide one final product to end customers and implement chain management functionalities is called a Multi-agent System for Supply Chain Management (MASCM). In this chapter, we first introduce the basic ideas of supply chain and chain management from an operational economics perspective. Then we clarify concepts of MASCM, many of which are directly or indirectly linked to the concepts behind traditional supply chain and chain management.

2.1 Enterprises and a marketplace

An enterprise (organization or firm) is a basic economic unit that produces a product (good or service) for sale. A marketplace (market) is a group of enterprises and human beings interacting with each other to buy and sell under certain controlled conditions. The purpose of all activities that an enterprise performs in the marketplace is to earn maximum profits for its owner or shareholders over a long time period [1]. It has been observed that, in an uncertain world the concept of “maximum profit” is not clearly defined. While a particular course of action might not result in a unique, deterministic but a variety of possible levels of profits, each of which has a certain probability of occurrence. It is meaningful to say that a firm is pursuing maximized profit only when the firm can explicitly or implicitly attach a probability to each level of profit that could result from each course of action. The time period for measuring profit must also be defined. In general, it means that the enterprise will not count on the profits from current, limited number of transactions but the sum of profits from the ones over a relatively long time period such as one year. In the real life, an enterprise’s long-term profits-earning targets might conflict with short-term ones. For instance, increasing the price of a product could temporarily increase a retailer’s revenue but could reduce the number of. If the gain from the increasing price cannot cover the loss from the decreased number of customers, the enterprise may reduce long-term profits, e.g. the expected revenue for a financial year. Therefore, it can be assumed that, if the result of an action can be predicted, an enterprise will always sacrifice short-term benefits for long-term goals. In a market, an enterprise usually has the following attributes. First, it is a rational entity. That is, it knows how to rationally choose a suitable action from a set of alternatives. For example, hungry buyers know that they should choose foods rather than clothes when they shop in a market. Second, an enterprise's business behaviors are consistent. In other words, its actions can be determined and, to some extent, predicted by a finite set of rules. Third, each enterprise is greedy and self-interested. It prefers more commodities to fewer given the same conditions such as price and quality.

A product is defined as a scarce and none-free valued resource that either can be used to produce other goods or directly satisfied human wants. With respect to a particular product, the resource provider side is called a product supplier or just supplier; the resources consume side is called a product consumer or just consumer. Goods that are directly consumed by the human beings are called *final products.* Human beings are called *end customers*. The supply-demand connection is the basic and fundamental relationship between enterprises. All other relationship can be directly or indirectly derived from it. Furthermore, a market, consisting of suppliers and customers, can be classified into four categories, *perfect competition*, *monopoly*, *monopolistic competition* and *oligopoly*, according to the number of product suppliers [1]. If the market, for any product, has many suppliers, it can be classified as a perfect competition or monopolistic competition marketplace. If there is only one supplier for a certain product, the market is a monopoly. The oligopoly is the intermediate case, in which a product only has a few suppliers. In this dissertation, we discuss the perfect competition market with the following attributes:

1) Any supplier of a given product can provide the same service as anyone else. That is, consumers of the product do not care whether they purchase the product from one seller or another, as long as the price is the same and the product is available.
2) No single supplier or customer can affect market attributes related to a product such as the price and where and how it goes.
3) All enterprises can join and leave the market freely.
4) Both supplier and demand have a perfect knowledge of product data. This information includes the name of the enterprise, the product requirements and good description.

In a perfect competition market, no central authority controls transaction between the demand and supply sides. Since both sides are greedy and self-interested, there are no existing agreements for certain product allocation. The conflicting interests are resolved through *negotiation process* that might lead to a compromised solution somewhere between each side's ideal outcome and one barely accepted. The successful negotiation process results in a *commitment* that can be a contract or just a verbal promise. In either case, a neutral party trusted by both buyer and supplier monitors the commitment. If one side does not honor the commitment, it can be punished either explicitly through confiscation of the transaction deposits or implicitly by receiving a bad reputation.

2.2 The concept of supply chain and its management

In this section we introduce the basic concepts of supply chain and its management. A general definition of supply chain can be found in [13]. The definition is "A supply chain is a network of facilities that procures raw materials, transforms them into intermediate subassemblies and final products and then delivers the products to customers through a distribution system." A supply chain is commonly referred to as a network because it involves the bi-directional flows of materials, information and money. The information flow is defined as enterprises' interactive actions in electronic, spoken or written format. The material and money flows are defined as products and payment transfer between enterprises, respectively. In this dissertation, the term "supply chain" is used interchangeably with the term "supply chain network (SCN)" or "supply chain system."

A supplier network, also called an *upstream network* or just an *upstream*, consists of all organizations that provide inputs, either directly or indirectly to the focal firm. Similarly, the customer network of an enterprise is called *downstream network* or just *downstream*. For example, an automotive company's upstream includes the thousands of firms that provide items ranging from raw materials such as steel and plastics, to complex assemblies and subassemblies such as transmissions and brakes. The upstream may include both internal divisions of the company as well as external suppliers. A given material may pass through multiple processes within multiple suppliers and divisions before being assembled into a vehicle. The vehicle may go through its downstream, consisting of warehouses, long distance transportation companies and dealers, and finally reaches the end customer. A supplier for this car company has its own set of suppliers that provide inputs that is also part of this supply chain. The beginning of a supply chain inevitably can be traced back to "mother earth"; that is, the ultimate original source of all materials that flow through the chain (e.g., iron ore, coal, petroleum, wood, etc.). A supply chain is essentially a series of linked suppliers and customers; every customer is in turn a supplier to the next downstream organization until a final product reaches the ultimate end user. We always assume a supply chain is in a “perfect competition” type of market.

Enterprises with no direct supply-demand relationship but that share or partially share a set of direct customers are said to sit at the same *tier* as each other. Direct suppliers of an enterprise are one tier level higher that. The tier an enterprise is in shows the relative importance of supply inputs to the enterprise. Roughly speaking, the higher tier an enterprise belongs to, the less impact supply inputs has on its ability to complete orders. However, since the supply-demand relationship can be complex, with different business connections, an enterprise can be placed in more than one tier. Only by adding additional constraints can the exact tier of each enterprise in a supply chain be determined. At that time the chain can be described as a “leveled” network.

A graphic illustration of a supply chain follows,

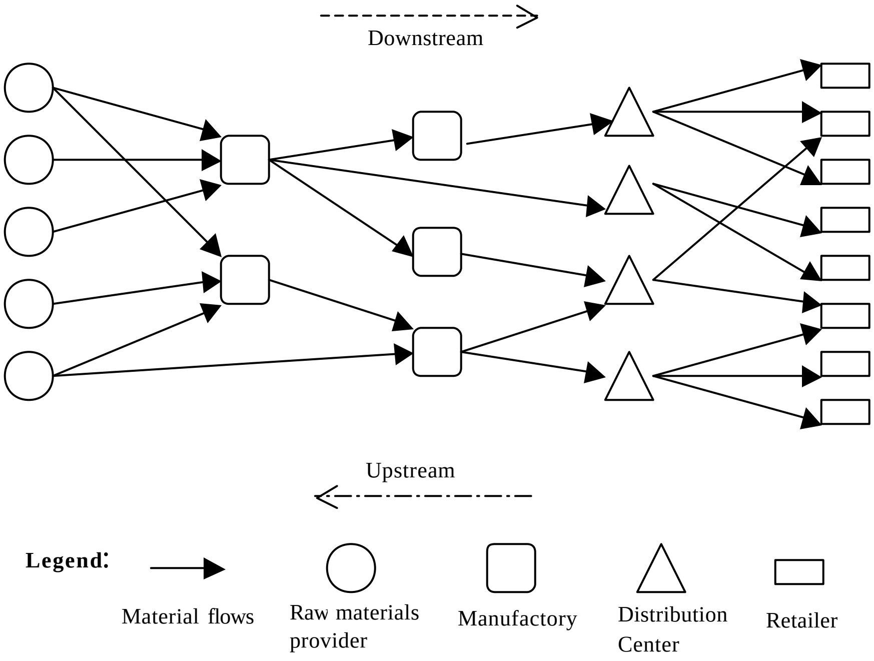


Figure 2.1 An example of supply chain

2.2.1 Supply chain classifications

Supply chains exist in virtually every industry, especially industries that involve product manufacturing. In [14], Lin identified three main types of supply chain networks, Type I, II and III, based on such attributes as manufacturing process, primary business objective, product differentiation, range of product variation, assembly stages, product life cycle, and main inventory type as shown in the following table.

Attributes	**Type I SCN**	**Type II SCN**	**Type III SCN**
Manufacturing process	Convergent Assembly	Divergent Assembly	Divergent Differentiation
Primary business objectives	Lean production	Customization	Responsiveness

Product differentiation	Early	Late	Late
Range of product variations	Small	Medium	Large
Assembly process	Concentrating at the manufacturing stage	Distributed to the distribution stage	Concentrating at the manufacturing stage
Product life cycle	Years	Months to years	Weeks to months
Main inventory type	End products	Semi-products	Raw materials
Example industries	Automobile and aerospace	Appliance, electronics and computers	Apparel/fashion

Table 2.1 Different types of supply chain

The automobile and aerospace industries are associated with Type I supply chain networks, where two main issues are how to efficiently meet customer demand without carrying excessive inventory, and how to coordinate suppliers and assemblers to smooth material flow. Structurally, there are many suppliers. The wide range of materials and sub-components that come from these suppliers converges through a series of manufacturing stages until the final product is assembled at one location. The final product is then shipped to several distributors and ultimately to a large number of retailers. The appliance, electronics, and computer industries can be classified as Type II supply chain networks, where the main issues are reducing the lead-time (planning, scheduling and manufacturing time) of the assembly-to-order process, and managing the inventory and purchasing for the assembly. In these supply chain networks, a relatively small number of suppliers provide materials and sub-components that are used to produce a number of generic product models. Complex assembly processes for generic models (semi-products) are executed at factory sites, and simple assembly processes for customized models are executed at distribution sites. A number of distribution points may be needed to quickly respond to customized orders. The apparel/fashion industry is a Type III supply chain, where the main issues are acquiring market information to respond to demand, and deferring product differentiation to maintain flexibility to handle constantly changing markets. In these supply chain networks, the number of end items is larger than the number of raw materials. There are a small number of suppliers and manufacturers, but a larger number of distributors and retailers. These three types of supply chains serve as the basis for understanding the issues and challenges for improving supply chain management.

2.2.2 An abstract of a supply chain and its experimental research foundation

Any business behavior of an enterprise or a human will create one or more flows to pass through the supply chain network. Among the three types of flows (information, material and money flows), the information flow is in a dominant place because it usually defines the attributes of the other two types and affects their creation and consumption. For example, the phone conversation begins and a contract is created before the product delivery. The contract or commitment records how the product is to be transported from supplier to customer and how much the customer must pay. In other words, in a supply chain, once information flows are determined, the chain status is set. From this perspective, a supply chain can be defined by the enterprises and the information flows it contains. This observation gives a high level abstract of supply chains. It also provides the foundation for supply chain experimental research. In this dissertation, we study a supply chain system based on this abstract.

Within a supply chain, information flows can be roughly classified into two categories. One type of information flow is used to establish a demand-supply relationship between enterprises. This type of information flow includes supplier advertisement, the process that a customer uses to search for suppliers, and the negotiation processes that might lead to a commitment among negotiating parties. The negotiation process is the most important information flow of this type. In the real life, in order to save inventory cost, an enterprise usually starts its internal processing once it receives an incoming request. This is a very common business strategy, called Just In Time (JIT), which has been widely used and studied. When this strategy is used, it is the customer side that initiates the negotiation process. If not specified otherwise specified in the dissertation the term “negotiation” refers to JIT with the customer side triggering the process. The other type of information flow involves commitment processing, which maintains the supply-demand relationship between enterprises. Commitment processing is the core of a demand-supply relationship; that is, the demand side satisfaction is relied on supply side performance. This dependency between supply side and demand side is the basis for modeling and analyzing business interactions in a supply chain.

2.2.3 Supply chain management

The activities of an enterprise in the supply chain include sourcing and procurement, production scheduling, order-processing, inventory management and so on. They can be roughly divided into three categorizes:

- Internal functions,
- Upstream management
- Downstream management.

Supply chain management of an enterprise is the integration of these three activities [4].

An enterprise's internal functions include the different processes used in transforming the inputs provided by its upstream to the output requested by its downstream. In the case of an automotive company, this includes all of its parts manufacturing (e.g., stamping, power train, and components), which are eventually brought together in their final assembly operations to create actual automobiles.

The two most important internal functions are order processing and production scheduling. Order processing is responsible for translating customer requirements into actual orders. It may involve extensive customer interaction, including quoting prices, possible delivery dates, delivery arrangements and after-market service. Production scheduling translates orders into actual production tasks. This may involve working with Materials Requirement Planning (MPR) systems, scheduling work centers, employees, and maintenance on machines.

Another important enterprise activity involves the management of upstream external supply chain members. In order to manage the flow of materials between all of the upstream organizations in a supply chain, firms need to ensure that the right materials arrive at the right locations, at the right time. These activities include:

- Supplier selection

- Supplier performance monitoring and evaluation
- Employment of appropriate contractual mechanisms
- Relationship maintenance.

Lastly, a firm has to take care of the downstream that encompasses all of the distribution channels, processes, and functions so that the product passes through its way to the customer. An enterprise may have relatively small upstream but fairly long downstream distribution channels. The downstream management includes

- Issuing information related to current situation about the commitment processing
- Dealing with the actual movement of materials between locations
- Collecting payments and feedback.

2.2.3.1 The performance measurement of supply chain management

For a given enterprise, supply chain management is not an easy task because large numbers of activities must be coordinated across organizational and global boundaries. The goal of supply chain management is to balance the internal operations, requests from customers, and supplier performance so that the enterprise successfully completes commitments (contracts) and earns profits. The goal is also described as to deliver "*the right products in the right quantities (at the right place) at the right moment at minimal cost*." (From NEVEM-workgroup [15]). More specially, the supply chain management tries to achieve following targets [16].

Customer satisfaction

Improvement in customer satisfaction can enhance an enterprise’s reputation and lead to new business opertunities. Typical measures of customer service are a company's ability to fill orders within due date (fill rate) and its ability to deliver products to customers within the time quoted (on-time deliveries). Other metrics should be used to evaluate the delivery performance of orders that are not delivered on-time. One way is to measure the average time from order to delivery.

Minimizing inventories

Manufacturing entities have inventories for raw products (RPI), products in the production process (WIP), and finished products (FGI). In addition, there are often warehouses or distribution centers between the different levels of the downstream. Inventories are costly. Binding capital in inventories prevents the company from investing this capital in projects of higher return. The holding cost inventories are therefore often set as high as 30 - 40% of the inventory value. It is in every enterprise's interest to keep inventory levels at a minimum.

Flexibility

Flexibility can be defined as the ability to respond to changes in the environment. In the case of a manufacturer, flexibility is the ability to change the output in response to changes in the demand. A flexible enterprise can capture market share more readily. Enterprises usually rely on *safety stocks*, which reserve certain amounts of product or service capbilities to improve their flexibility. Upstream enterprises’ flexibility can affect flexibility of their direct or indirect customers.

The performance for an enterprise’s supply chain management can be measured by whether it can attain one or more of the above goals. However, there are usually tradeoffs between the different goals. It is difficult to meet all of them simutaneously. For example, if lead time is constant, the ability to fulfill orders is directly dependent on the inventory levels in a supply chain. As long as there are products in the Finished Goods Inventory (FGI), from which products are taken, orders can be satisfied. But oversized inventories is costly. On the other hand, the company's reputation may be severely damaged if it can not complete the customer orders within the required time. Therefore, the enterprise will find it to achieve both minimum inventory holding cost and perfect customer satisfaction. The trade-off between inventory costs and customer satisfaction is one of the classic issues of logistics and supply chain management.

From a system perspective, supply chain management is a management process that attempts to optimize the operation of the entire supply chain. Although different entities in a supply chain typically operate subject to different sets of constraints and objectives, there is one central, overriding focus toward supply chain management for all the enterprises in the chain, that is, continual improvement of end-customer service. [17]. All the enterprise will profit by receiving more orders from their direct customers if the number of end orders increases. End customer needs must be satisfied if overall supply chain is to succeed on a long-term basis. Thus, at the system level, the fundamental concern of supply chain management is to continually reduce total cycle time and improve the efficiency of end customer order fulfillment [17]. In other words, the overall supply chain management performance can be measured by end customer satisfaction.

Beyond this customer-oriented aspect of effective management performance measurement, a number of phenomena indicative of overall supply chain management desirability have been stressed. They are:

- Changes in both the average volume of inventory held and frequency turns across the supply chain over time [18].
- The adaptability of the supply chain as a whole to meet emergent end customer needs [19]
- The extents to which relationships between chain members are based on mutual trust [19].

In this dissertation, the overall supply chain management performance is measured by end customer satisfaction. The better the customer satisfaction, the better the performance of overall supply chain management.

2.2.3.2 Uncertainty in supply chain management

One major problem involved in supply chain management is understanding and managing the uncertainties. This is especially true in industries such as fashion ski-wear where demand is heavily dependent on a variety of factors that are difficult to predict - weather, fashion trends, the economy - and the peak of the retail selling season is only two months long [20]. Three fundamental sources of uncertainty exist along a supply chain. They include demand (volume and mix), process (yield, machine downtimes, transportation reliabilities), and supply (part quality, delivery reliabilities) [21] [22]. The source of uncertainty in the supply chain will be further discussed in chapter 3. It has been indicated by previous studies that through improved collection and sharing of information between supply chain members, uncertainty can be well-managed [23]. Information sharing also results in better customer service, through better coordination, and improves asset management, by giving decision-makers the information necessary to optimize inventory and capital asset costs. The difficulty arises when trying to design an information sharing mechanism that can handle the information needs of each of the supply chain members to allow efficient, flexible, and decentralized supply chain management in a dynamic and uncertain environment. Three approaches address this problem; they are case study, simulation and formal modeling. Currently, there is no model that can be used to generalize and analyze uncertain information-sharing for a supply chain management system [33]. In Chapter 4, we introduce the extended Bayesian Belief Network (eBBN) as formal model to study chain member interactions in the uncertain setting.

In summary, a supply chain is an enterprise network that delivers only one type of final product to end customer. Chain members are linked through information, material and money flows. The overall supply chain management goal is to maximally satisfy the needs of the end customer. Dealing with uncertainty is an important challenge for supply chain management but it has not been adequately addressed.

The description of a supply chain and its management is summarized in following table.

1. Supply members (enterprise)	Suppliers and customers (includes manufacturers, assemblers, distributors and so on)
2. Chain flows	Material, payment and information flows
3. Interdependencies	Demand-supply relationship that includes material shipments and orders, funds transfer, and information sharing.
4. Supply chain management goals for a given enterprise	Minimize order fulfillment cycle time, Minimize inventory levels and costs, Maximize flexibility
5. Overall supply chain management measurement	End customer satisfaction
6. Problem addressed	Study uncertain information sharing among enterprises in a supply chain

Table 2.2 Supply chain and its management

2.3 A framework of a Multi-agent System for Supply chain Management

In the last decade, computing power has increased dramatically. The CPU speed of a computer doubles every two years with little change in price. In the mean time, the popularity of new generation programming languages such as Java makes commercial software development easier and faster. Nowadays, powerful software and hardware are affordable for many enterprises, allowing them to adopt computer technologies to save operating expenses and improve efficiency. The development of networks and information sharing allow enterprises to establish close connections through electronic data exchange and automatic procedure handling. This capability has also lead to the emergence of electronic commerce (e-commerce), defined as "doing business electronically" [24]. With these developments, supply chains gradually have evolved into autonomous systems. The technologies that are relevant to our discussion include Internet, Electronic Data Interchange (EDI), and Extensible Markup Language (XML)/EDI.

With the proliferation of PCs, LANs, and modems and the establishment of open standards such as TCP/IP, HTTP, and HTML, the Internet has become the system that allows information sharing among supply chain partners across geographical regions. Originally, Internet was developed to be a pool of human knowledge that would allow collaborators in remote sites to share their ideas and all aspects of a joint project [25]. Because a supply chain is similar to the projects the Web was designed (remote sites, shared knowledge, common target), the Web can serve as an infrastructure for sharing of information in a supply chain.

To permit automatic and electronic date exchange, information must be structured according to predefined formats and rules that a computer can use directly [26]. EDI is an existing information technology that provides a method of electronic business-to-business transaction transfer between computers without interpretation or transcription by people. EDI technology was shown to facilitate accurate, frequent, and timely exchange of information to coordinate material movements between trading partners. It increases the speed and the accuracy of processes compared with non-electronic transfer of information [27]. When a supplier and a procurer use information technology to create joint, interpenetrating processes at the interface between value-adding stages, they are taking advantage of the electronic integration effect. Benefits include is the time saved and the errors avoided because data need only be entered once [28].

A practical problem needs to be addressed EDI is that it lacks of a globally recognized standard format for data storage and transfer across the Internet [27]. One solution that has been considered by a number of businesses is to use the combination of Extensible Markup Language (XML) and EDI. XML is the subset of Standard Generalized Markup Language (SGML) developed by the World Wide Web Consortium (W3C). XML allows users to specify the role and syntax of each piece of data in an interchanged document and the order in which each piece of information is expected. XML also identifies which programs should be used to control the document exchange. It can encode the document's information precisely and in a richer structure than was previously possible with earlier formats [29]. Through this way, XML/EDI transactions are self-describing and can be automatically processed by applications over the world.

With the growing maturity of the technologies we discussed above and with the rapid development in information technologies, it can be seen that an entity (enterprise) in the supply chain can make decision and interact with others electronically and automatically. In [6], we present a framework of electronic supply chain management system, called Multi-agent System for Supply Chain Management (MASCM). It is described as follows.

MASCM is an electronic marketplace.

MASCM is a special perfect competition marketplace. The participants in this marketplace are virtual enterprises, called software agents, which conduct business actions using software applications and electronic data flows with little human intervention. Business entities are allowed to join or leave this market freely. However, the marketplace only provides one final product to end customers.

MASCM is a chain management system.

A supply chain in MASCM consists of software agents connected by supply-demand relationship. MASCM provides basic mechanisms for software agents to pursue their various chain management goals in business activities. These mechanisms include social conventions for conducting business and other functions that facilitate transactions among software agents (naming service, ontology definition etc.). In a MASCM, there can exist one or more supply chain at the same time. The components of the supply chain can change over time, however, the system ensures at any given time all supply chains pursue one and only one common goal as their overall chain management target, e.g. the end customer satisfaction.

The following figure shows a graphic illustration of a MASCM.

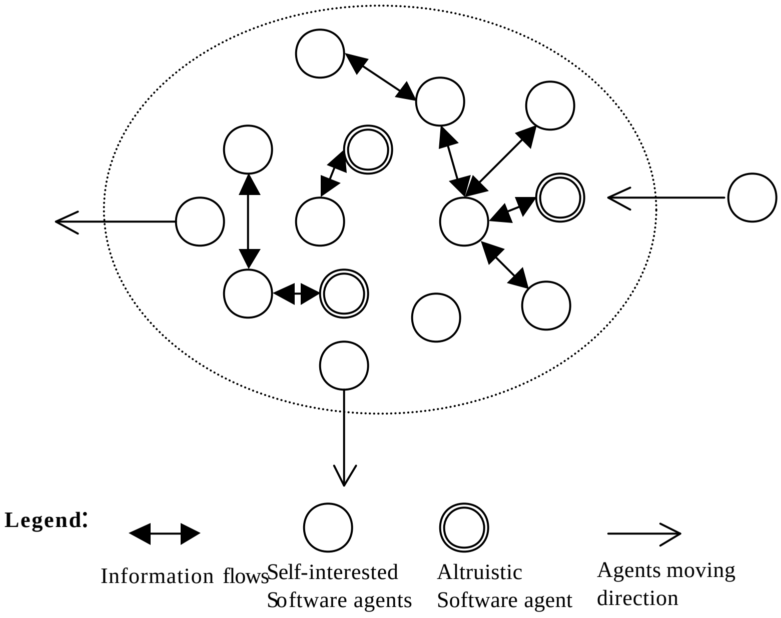


Figure 2.2 An illustration of MASCM

In following sections we discuss MASCM in detail.

2.3.1 Software agents as virtual enterprises

Although there is still no universal agreement on the definition of a software agent, many researchers define software agents with following attributes: autonomy, learning and cooperation [5]. Autonomy refers to the principle that agents can operate with little or no human guidance. The key element of autonomy is proactivity. That is, when given a goal, a software agent knows how to take actions to reach it. The attribute of learning means software agents can learn from the experiences and knowledge they gain from interaction with humans or other computational entities. This learning ability is critical criterion for distinguishing a software agent from ordinary software. One simple measurement of whether a software agent has learning ability is to see if it can automatically improve performance over time [30]. Lastly, the attribute of cooperation, which is paramount, describes agents taking on roles in an artificial society e.g. MASCM instead of standing alone.

Software agents used to represent virtual enterprise must also be business-oriented and rational. This means their behaviors can be described by a finite rule set of business logics. These rules guild agents to approach goals set by their owners.

Based on their business behavior in a marketplace, software agents in MASCM fall into two categories:

Functional agents. These types of virtual enterprises buy or sell products in order to make profits for their owners. These agents are self-interested, and designed and implemented by different agent owners. In the system each is one of the suppliers to the end customer. We also assume each functional agent sells one and only one product in a MASCM.

Informational agents. These agents provide public service including supplier (customer) lookup, commitment verification and justice, system registration etc…. Informational agents are altruistic and do not belong to particular owners. As the party trusted by every functional agent in the system, informational agents can be defined as the components of a specific MASCM. They are maintained by the organizer of marketplace. Their operational expenses are collected by levying a “tax” on all functional agents.

2.3.2 Information flows

Both types of information flows discussed in Section 2.2.2 have to be implemented in a MASCM. They consist of sequences of messages exchanged between agents. In the future, Information flow is interchangeable with the term "message". Messages used to establish the demand-supply relationship are more much complicated than ones used to maintain the relationship. The first case includes the query messages from functional agents to informational agents and the negotiating messages between two functional agents. The exchange commitment status between two functional agents is an example of the second case.

Usually, the study of information flows in a MASCM should include the following areas,

Communication. How agents exchange messages. Communication requires: an interaction protocol, a communication language, and a transport protocol. The interaction protocol refers to the high level strategy, pursued by the agent, that governs its interaction. Such a protocol can range from negotiation schemes and game theory protocols to a simple one such as "every time you do not know something, find someone who knows and ask." The communication language is the medium through which the attitudes regarding the content are communicated. The transport protocol is the actual transport mechanism used for the communication, such as TCP, SMTP, HTTP, etc.

Representation. How agents represent complex objects (physical or abstract). This may require the support of a sophisticated representation scheme and ontology. Examples can range from orders to contracts.

Problem solving. How agents extract the content from information flows, and how they apply their reasoning to it. There is a large body of work on algorithms and techniques for constraint solving that can be applied to this problem [31].

Human interaction. How the information flows can be integrated with humans in appropriate ways, either for authorization and authentication or as part of a larger workflow environment.

Communication is a fundamental issue for transactions between agents. It is largely domain-independent in that the nature of the content being transferred does not matter. Representation, problem solving, and human interaction are problem- or agent-dependent and require detailed consideration when a system is being constructed. In the dissertation, we assume:

- Agents use Knowledge Query and Manipulation Language (KQML) as their communication language.
- Interaction protocols and representation schemes are well known by all entities.
- Agents use proper decision-making algorithms chosen by their owners to retrieve and utilize the information sent by other parties.
- Agent owners have a firm control and understanding of agent actions in the system.

The detailed study about KQML is in [32]. A high-level negotiation protocol for a MASCM and its analysis can be found in [6].

2.3.3 Order Fulfillment Process (OFP)

Since JIT is widely used as business logic for many enterprises, we assume it is the default strategy functional agents follows. Correspondingly, chain activities described in Section 2.2 that a software agent is involved in can be further simplified as an *Order Fulfillment Process* (OFP) [34]. Thus, functional agent' behaviors in a supply chain can be logically divided into the following steps.

1) **Order generation.** Based on the commitment made to its customers or set by internal needs, the functional agent selects suppliers, generates orders and chooses negotiation strategies for each supplier. Since more than one supplier is usually needed to provide different materials this functional agent may generate more than one order to fulfill the commitment. No information flows are triggered at this time.

2) **Negotiation.** The functional agent sends orders to the desired suppliers and negotiates with them. The functional agent can negotiate simultaneously with different suppliers. However, we usually assume in each negotiation process, there is only one supplier is involved. That is, the default negotiation protocol between two functional agents is bilateral. An order is *temporarily solved* if there is a commitment reached with a supplier. If there is no agreement, the

functional agent must look for an alternative supplier or must fail in its commitment. Information flow involved in this step is used to establish an active instance of supply-demand relationship, called a *sell-buy connection*.

3) **Commitment processing.** The functional agent processes the commitment, deals with the unexpected events, and exchanges information with its supplier and customers. When it receives notice that the product is delivered by a supplier, the order sent to this supplier is *eventually solved*, and the commitment between these two agents has been *resolved successfully*. An order will not be eventually solved if its supplier aborts the commitment or itself cancels the commitment following the decision made by internal functions. The consequence of commitment cancellation by this functional agent is to cause all direct supplier agents involved in sell-buy connections with that functional agent to cancel any orders they have generated which have not yet been eventually solved. When direct suppliers provide all the materials the functional agent needs, the commitment held by this agent is also resolved successfully. We say an OFP initialized by this functional agent c*ompletes*. Otherwise, the commitment is *resolved unsuccessfully* and an OFP *fails*. In either case, the functional agent has to notify its customers. Information flows involved in this step are to maintain the active supply-demand relationship or buy-sell connection.

On thing to be noticed is that the steps above are logically divided according to type of information flows. A functional agent can be involved in multiple and different type of information flows at the same time. Therefore, in practice, an OFP cannot be actually divided into sequential steps. For example, let us consider a case in which a functional agent makes commitments with more than one supplier, but suddenly one of them abandons the commitment. This agent then has to search for an alternative supplier and negotiate with it while exchanging information with other suppliers. During that period, we cannot tell at which step of an OFP this functional agent is.

We assume that each functional agent has a certain inventory of the products it needs or safety stock before it makes a commitment to customer. In other words, suppliers might have full or partial capabilities to fulfill the customer's order without asking for help from their suppliers. This assumption assures that a functional agent has some flexibility in responding to customer requests. Thus, customers have freedom to choose or switch suppliers.

When its activities in a supply chain are described as an OFP, a functional agent controls each step of an OFP properly to reach its chain management goals.

2.3.4 Virtual supply chain and system performance

Instead of being formed several rigid networks in the marketplace, the supply chain in the MASCM is constructed dynamically [6]. That is, when the end customer submits an order, called an end order, to end customer agent (functional agent), this special order will trigger interconnected OFPs. All functional agents involved in OFPs (related to that end order) with supply-demand relationships comprise a dynamic supply chain. This supply chain is called a Virtual Supply Chain (VSC). If the end customer agent's OFP completes, all functional agents that have resolved commitments successfully in the serial of OFPs with the supply-demand relationship among them, define a *completed* VSC. It is the solution to the end order. If an OFP triggered by end customer agent fails, the system cannot find a solution to the end order. Correspondingly, a VSC that emerges after an end customer commitment is created but before it is resolved either successfully or unsuccessfully is called an *evolving* VSC. In a MASCM, at any given time, there might be more than one VSC. And a functional agent might be a participant in different VSCs. The following figure illustrates a VSC emerging when an end order arrives in a system.

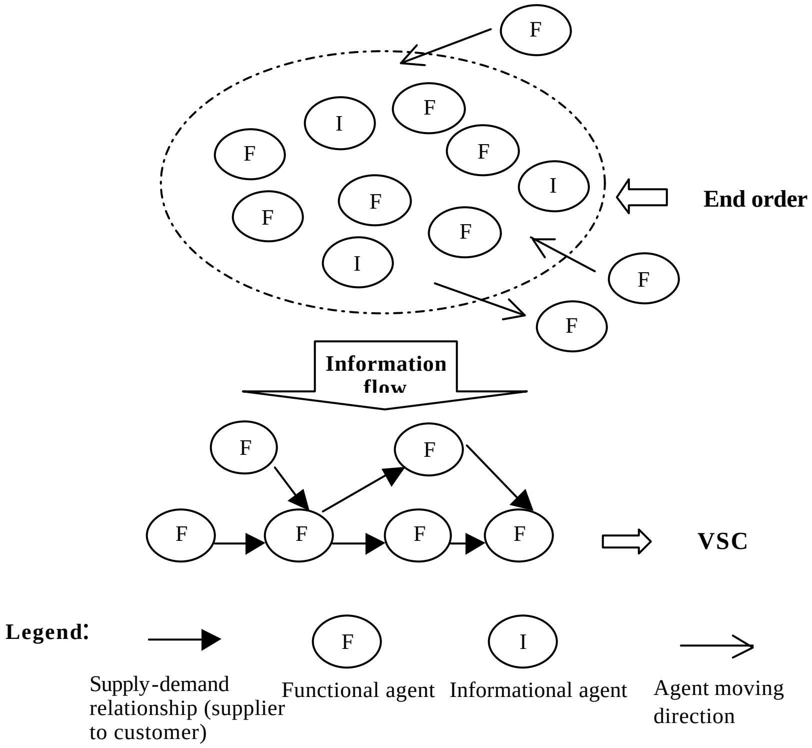


Figure 2.3 An end order and a VSC

Since all VSCs in a MASCM shared one overall chain management goal, this common target is used to define as the MASCM management target. Performance of a MASCM can be measured by checking whether this target has been reached or not. In this dissertation, the MASCM management goal is defined as end customer satisfaction. Thus, related metrics, e.g. the system responding time and the number of end customer orders with solutions can be used to measure or compare system performance.

2.4 Summary

In this Chapter we reviewed the concept of supply chain and chain management and discussed the Multi-agent System for Supply Chain Management (MASCM). Agents' activities in a supply chain have been described as an Order Fulfillment Process (OFP). To any given order, through interconnected OFPs, a Virtual Supply Chain may emerge. System management goal is defined as end customer satisfaction. The system performance is measured by checking whether this goal has been reached. The MASCM is the framework for this dissertation research.

Chapter 3 Uncertainty in a MASCM and Bayesian Belief Networks

In this Chapter we discuss uncertainty in a MASCM, and its impact on supply chain management and system performance. We also review the basic concepts of Bayesian Belief Networks that serve as the fundamental model for our future studies.

3.1 Uncertainty in a MASCM

The term "uncertainty" in an agent system is used to describe the fact that agents' behaviors cannot be known with certainty before they actually take place. As many researchers point out, to study the commitment between software agents is fundamental, it is fundermental to understand the behavior of an agent that takes a social role in a multi-agent system.

In a MASCM, each software agent takes a role as a product producer or consumer. Their actions are not isolated but interactive ones. The major and important interactions occur among functional agents, in which they communicate, negotiate and share information so that they can sell or buy products to gain profits for their owners. A commitment is the common goal for social activities of functional agents in the supply chain. It works as a mutual agreement for all parties in the interaction. Given a pair of functional agents from different tiers, one acts as a product supplier or consumer direct or indirectly to the other one. A supplier can determine from the commitment what is the particular goal to pursue so that it can design and execute the internal action plan. A customer, based on the commitment, can evaluate upstream works and adjust its corresponding interactive strategies. A commitment also can coordinate behaviors of functional agents at the same tier to achieve extra benefits, for example, gain monopoly price over a single resource (product) control. In all these scenarios, commitment plays the central roles in functional agents' autonomous actions. Consequently, a functional agent's unpredictable behaviors will eventually reflex in its commitment execution process. In other words, in the MASCM, the type of uncertainty being concerned is how likely an agent is to break the promise it has made to others or the possibility of a commitment to fail.

In this section, we discuss the source of uncertainty that could make a commitment fail and its impact on supply chain management.

3.1.1 The sources of uncertainty

Uncertainty comes from following three sources.

First, it can from functional agent's limited and local knowledge about the system. To complete an Order Fulfillment Process (OFP), a functional agent has to rely on its capability as well as inputs from suppliers. It is impossible for a functional agent to collect all the information on its upstream suppliers before it makes commitment to the customer. The commitment is always made with performance estimation of its suppliers.

When the system allows the agent to join and leave it freely, the global information becomes even more difficult to gather. If an agent make an incorrect guess of its supplier, its commitment is likely to be retracted later [35].

Second, it can from agents' strategic self-interested actions. Each functional agent takes actions on behalf of its owner. In order to protect its own interests or gain extra benefits, a functional agent may choose to issue a false, overoptimistic or over-pessimistic claim related to the incoming order request. The commitment based on such information is fragile and easy to fail. Compared with cheating actions in a traditional supply chain this type of behavior if much more prevalent in a MASCM. In traditional supply chains, human beings involve in each step of the procedure to set up a business relationship including look up a service provider, negotiate and make commitment. Human involvement leads to higher investment and easier cheating catches. As a result, in traditional chains, a dishonorable action is thought as the high risk with low reward and irrational decision with little attempt. On the contrary, in a MASCM, all business processes get through the automatic information exchange procedures with relatively low operational cost and less misconduct discovery possibilities. This makes agents to be inclined to seek one-shut intangible benefits.

Third, it can come from unpredictable events in marketplace or internal components of a functional agent. For example a network jam might cause two agents to lose communication for a long period and will be treat as a de-commitment by one side. Bad weather may delay the product delivery. Other events include computer crash, sudden human intervention or shortage of labor resources and so on. These events are uncontrollable by any single functional agent. Even though most of them are rare events, they will make prediction of other functional agents' behaviors more difficult.

3.1.2 The impact of uncertainty on supply chain management

As we discuss in last chapter, the goal of MASCM management work is to ensure the demand from end customers can be mostly satisfied. Thus, the system performance can be measured by shorted processing cycle per end order and higher percentage of the ender user order fulfillment. For different functional agents, they may emphasize different aspects of chain management, e.g. minimum inventory and so on. However, the impact of uncertainty makes negative impacts to both system performance and management target of individual functional agents.

When there is an end order put into system by end customer, an interconnected OFP may emerge. If there is no uncertainty, functional agent can exact know what it can do for its customer, the process to construct a VSP can be simplified as a Distributed Satisfaction Problem (DSP) [31] that the order from end customers is solved by a solution consisting a serial of sub-solution contributed by functional agents within their restrictions. In this setting, it is usually not necessary to search all possible combination of different functional agents in the system or at most once to know whether there is a solution or not. With uncertainty, the searching process becomes complicated. Since a functional agent may get wrong estimation of its suppliers, and unexpected events can let it be adjusted again and again, a solution to end order may be found out through many times of searching over all possible combinations. In addition, with uncertainty, the functional agent can fail in commitment execution (withdraw a commitment) because of many unpredictable events, it obviously increase failure possibility of the end order eventually. Compare the case with and without uncertainty, it can see that with uncertainty, end order process time is prolonged and its fulfillment possibility is lower. Accordingly, end customer satisfaction is reduced. In other words, uncertainty can damage system performance.

To an individual functional agent, when an unexpected event related to the demand-supply connection occurs, it is forced to change manufacturing, transportation, customer service plans and re-in-store all the produced products. It may also have to withdraw the commitment. Both of these actions can dissatisfy its customers since it fails to fulfill the task following the original agreement. Face the uncertainty, if functional agents choose to keep the customer satisfaction or flexibility as its priority, it have to keep a high volume "safety stock." This increases its inventory, so is the total amount of inventories in the system. No matter which strategy it chooses, uncertainty has negative impact on its management goal.

In realities, the uncertainty sources cannot be totally blocked. The impact from uncertain events cannot be ignored but counts heavily on supply chain goal accomplishment. Therefore, one critical task of MASCM design and implementation is to formally study functional agent interaction in uncertain settings and analyze the relationship between uncertainty management mechanism and system performance. As we can see the buy-sell activities among functional agents are triggered by end customer orders. These actions are connected link by link through commitments created in OFPs. That is, the commitment accomplishment possibilities are connected. The commitment-processing situation of a functional agent can affect the others'. For example, the failure of a commitment occurring at higher tier agents might increase the failure possibility of a commitment from the retailer agent to an end customer. From this perspective, functional agent interaction in the uncertain environment can be formalized as the propagation and analysis of commitment failure possibility through commitment network setup in the OFP after an end order put into the system. However, currently there is no serious research work on the modeling business entity behaviors related to uncertainty in the supply chain management system especially the emerging electronic chain management system. In the next chapter, we introduce a formal model, called extended Bayesian Belief Network (eBBN), which formalizes functional agent interaction related uncertainty and is used as analytical platform to study the impact of uncertainty on system performance. In the next section, we give a brief summary on Bayesian Belief Network (BBN).

3.2 Bayesian Belief Networks

Generally speaking, a Bayesian Belief Network (BBN) is a graphical model, which combines both probability and graph theory. Probability theory provides the glue whereby the parts are combined, and it ensures that the representation as a whole is consistent. The graph theoretic side of graphical models provides both an intuitively appealing interface by which humans can model sets of variables and their interdependence as well as a data structure that lends itself naturally to the design of efficient general-purpose algorithms. The graph itself consists of nodes and arcs. Nodes represent the random variables and the lack of arcs represents the assumption of conditional independence.

BBN has been proven a good tool to causal-effect knowledge representation and uncertainty reasoning. It initially arose from an attempt to add probabilities to expert systems and has a long history to be used in decision analysis [40]. One of the famous examples is a decision-theoretic reformulation of the Quick Medical Reference (QMR) model [36]. Nowadays, BBN model has been used as research analysis tools in the fields such as reinforce learning, speech recognition, tracking, data compression, etc. Practical applications include real time decision under uncertain situations [37], human-computer interaction analysis [38], deep-space exploration and knowledge acquisition [39], the popular productive software Microsoft Office, etc.

3.2.1 Basic axioms of probability

Probability theory, also known as inductive logic, is a system of reasoning under uncertainty. Within the Bayesian framework, probability is interpreted as a numerical measure of the degree of consistent belief in a proposition.

The probability of an event x (y), denoted by p(x) (p(y)), is a number in the interval [0,1], which obeys the following axioms,

- p(x)=1 if and only if x is certain.
- If x and y are mutually exclusive, then p(x or y) = p(x)+p(y).

We use lower-case letters to represent single variables and upper-case letters to represent sets of variables. Suppose x is a random variable having a finite number of mutually exclusive and exhaustive states $(a_1, a_2 ... a_n)$. Then p(x) will be represented by a vector of non-negative real numbers p(x)= $(b_1, b_2 ... b_n)$ where p(x=a_i)=b_i is a scalar and $\sum_i b_i = 1$.

A basic concept is that of conditional probability, a statement which takes the form: "Given y=c the probability of even x=a is b," written p(x=a|y=c)=b. It means if y=c is true, and other information is irrelevant to the x, then p(x=a)=b. The important property about conditional probability distribution is called *conditional independency*, that is, given random variable x, z and variable set Y if p(x|Y,z)=p(x|y), we say x and z is conditional independent given Y.

The fundamental rule for probability calculus is the product rule,

$$p(x \text{ and } y)=p(x|y)p(y) \text{ or } p(x, y)=p(x|y)p(y) \qquad (3.1)$$

Equation (3.1) defines the relationship among conditional probabilities, individual variable probability and joint probability for a set of variables.

The probability basis of BBN inference is Bayesian theorem as follows,

$$p(x \mid y) = \frac{p(y \mid x)p(x)}{p(y)} \tag{3.2}$$

Equation (3.2) can be easily obtained through algebraic manipulation of Equation (3.1). It can be interpreted as given a random event x, we have certain known knowledge (belief) represented as p(x), called *priori probability*. If a new event y is observed, the revised belief of event x, represented as p(x|y), called *posterior probability*, can be obtained by multiplying the prior p(x) by the ratio p(y|x)/p(y).

3.2.2 BBN representation

BBN provides a graphical representation for the joint distribution of a set of variables in terms of conditional and prior probabilities, in which nodes represent variables, and the orientations of the arcs represent influence between variables. Because variable and node are two different ways to represent a random event, in the future, these two terms are interchangeable. For example, the following diagram represents in different ways the joint distribution of random variable x and y. The first represents the prior beliefs (consider we know certain information of random event x) while the second represents the posterior beliefs (consider we observe a new random even y later). Usually, x is though as a possible "cause" of the "effect" y. The downward arrow represents this relationship. The upward arrow represents an “argument against the causal flow,” from the observed effect to the inferred cause. That is the reason the BBN is also called as causal network or influence diagram. In the context of this dissertation, these terms are interchangeable with BBN.

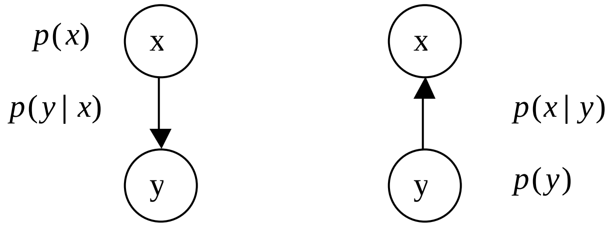


Figure 3.1 A simple example of using BBN to represent joint probability

Bayesian networks are generally more complicated than the ones in Figure 3.1, in the most general form, a BBN of n variables is a direct acyclic graph (DAG) of n nodes and a number of arcs, referred to as the network structure [40][41]. Nodes in a DAG correspond to variables, denoted x_i. Let a_i denote an instantiation of x_i, and $\{a_1, a_2, ...a_n\}$ representing a joint assignment or an instantiation to the set of all variables $X = \{x_1, x_2, ...x_n\}$. An arc $< x_j, x_i >$ represents a direct causal or influential relation from x_j to x_i. Also given in a BBN is the conditional probability distribution $p(x_i | p_i)$ for each variable x_i where p_i is the set of x_i’s parent nodes. When x_i does not have any parent, $p(x_i | p_i)$ becomes $p(x_i)$. The conditional independences assumption discussed above is made for belief: $p(x_i | p_i, Q) = p(x_i | p_i)$, where Q is any set of variables excluding x_i and its descendants. In this dissertation, we restrict all variables to be binary, i.e., $x_i \in \{0, 1\}$ ∀i.

We pay a special attention to a class of simpler BBN of binary nodes, known as the noisy-or network [40][41][42]. Instead of using the conditional probabilities $p(x_i | p_i)$, a noisy-or network associates a single probability measure, called *causal strength* and denoted c_{ji}, to each arc $< x_j, x_i >$. c_{ji} can be interpreted as the probability of $x_j = 1$, among the parents of x_j, to cause x_i to become 1. In addition to tremendously reducing the number of conditional probabilities needed to specify a BBN, the one-to-one correspondence between the arcs and causal strengths makes noisy-or networks more natural to humans than those using the more general form of the conditional probabilities $p(x_i | p_i)$.

3.2.3 BBN inference

BBN inference is the reasoning process over the uncertain knowledge stored over the graph structure. The computation of a joint probability distribution over variables $\{x_1, x_2 ... x_n\}$ can be determined based on the chain rule of probability, which is the extension of (3.1). It indicates that,

$$p(x_1, x_2 ... x_n) = \prod_{i=1}^{n} p(x_i | p_i) \tag{3.3}$$

For noisy-or networks, Equation (3.3) can be re-written as,

$$p(x_i = 1 | p_i) = 1 - \prod_{x_k \in p_i} (1 - c_{ki} a_k) \tag{3.4}$$

That is, $p(x_i = 1 | p_i)$ is a function of inputs a_k from the parent nodes x_k weighted with respective causal strength c_{ki}. Fro unspecified a_k, we have,

$$p(x_i | p_i) = x_i (1 - \prod_{x_k \in p_i} (1 - c_{ki} a_k)) + (1 - x_i) \prod_{x_k \in p_i} (1 - c_{ki} a_k) \tag{3.5}$$

Deduced from (3.3), in principle, any probability of interest in the domain can be computed from it. For example, suppose there is a very simple BBN with structure $w \to x \to y \to z$. If we want to know $p(w | z)$, it can be done through the following formula,

$$p(w | z) = \frac{\sum_{x,y} p(w, x, y, z)}{\sum_{x,y,w} p(w, x, y, z)} = \frac{p(w) \sum_x p(x | w) \sum_y p(y | x) p(z | y)}{\sum_w p(w) \sum_x p(w) p(x | w) \sum_y p(y | x) p(z | y)} \tag{3.6}$$

Besides the basic formulations similar to (3.6) discussion above, some other algorithms for probabilistic inference in BBN have been exploited. For example, Howard and Matheson [43], Olmsted [44] and Shachter [45] have developed an algorithm that reverses arcs in the network structure until the answer to the given probabilistic query can be read directly from the graph. Peal [46] has developed a message-passing scheme that updates the probability distributions for each node in a BBN in response to observations of one or more variables. Even though we can exploit assertion of conditional independence in a BBN for probabilistic inference, exact inference in an arbitrary BBN work is NP-hard [47]. Even approximate inference is NP-hard too [48].

3.2.4 Probability propagation

In the BBN, node probability can update locally via communication between vertices in the DAG. This allows the BBN can be used to model human reasoning. In [47], Pearl notes that such update scheme satisfies a primary goal of rule-base systems, namely the separation of the control mechanism or inference engine from the knowledge base. The scheme also can be easily implemented in object-oriented languages [42], which provides further advantages to model "belief" update and broadcast in a distributed system. In this section, we summarize Peal's probability propagation scheme in singly connected networks.

In the DAG, the probability update message from a parent node to a childe node is denoted as Π type of message; the message from a child node to a parent node is denoted as λ type of message. $l(x_i)$ and $\Pi(x_i)$ are called λ value and Π value respectively. They are the middle results used to compute $BEL(x_i)$ that represent the updated probability when node receives new information. Using the notation above and ones listed in Section 3.2.2, we have following propagation scheme that is adapted from [42].

Operative Equations

1. If x_i is a child of x_j, and it has another parent x_g, the λ message given x_i to x_j is given by

$$l_{x_i}(a_j) = \sum_{a_g=1,0} \Pi_{x_i}(x_g = a_g) \sum_{a_i=1,0} p(x_i = a_i \mid x_j, x_g)\, l(a_i)$$

2. If x_i is a child of x_j, the Π message given x_j to x_i is given by

$\Pi_{x_i}(a_i) = 1$, if x_i is instantiated for a_i;

$\Pi_{x_i}(a_i) = 0$, if x_i is instantiated as others than a_i;

$\Pi_{x_i}(a_i) = \frac{BEL(x_i = a_i)}{l_{x_i}(a_i)}$, if x_i is not instantiated.

3. If s(x_i) is the set of x_i, the λ value of x_i is given by

$l(a_i) = \prod_{x_k \in s(x_i)} l_{x_k}(a_i)$, if x_i is not instantiated;

$l(a_i) = 1$, if x_i is instantiated for a_i;

$l(a_i) = 0$, if x_i is instantiated as others than a_i.

4. If x_i has two parents x_j and x_g, The Π value of x_i is given by

$$p(a_i) = \sum_{a_j = 1,0} \sum_{a_g = 1,0} p(x_i = a_i \mid x_j = a_j, x_g = a_g)\Pi_{x_i}(a_j)\Pi_{x_i}(a_g)$$

5. The conditional probability of x_i based on the variable thus far instantiated, is given by,

$BEL(x_i = a_i) = a\, l(a_i)p(a_i)$ (a is normalization parameter)

The BBN is first initialized to compute the priori probabilities (i.e., the probabilities based on the instantiation of no variables) of all variables as follows:

Initialization

1. Set all λ values, λ message, and Π message to 1.
2. For all roots x_j, set $\Pi(x_j = a_j) = p(a_j)$, $a_j = 1,0$;
3. For all roots x_j for all children x_i of x_j, do

   Post a new Π message to x_i using operative formula 2.

{ x_j propagation flow will then begin due to updating procedure C.}

When a variable is instantiated, or a λ or Π type of message is received by a variable, one of the following updating procedures is used.

Updating

A. If a variable x_i is instantiated for a_i ($a_i = 1{,}0$), then

```
BEGIN
    1. Set $BEL(x_i = a_i) = 1$ and for $a_i^{'} \neq a_i$ ($a_i^{'} = 1{,}0$), set $p^{'}(a_i^{'}) = 0$;
    2. Compute $l(x_i)$ using operative formula 3;
    3. Post new $\lambda$ message to all $x_i$'s parents using operative formula 1;
    4. Post new $\Pi$ message to all $x_i$'s children using operative formula 2;
END.
```

B. If a variable x_i receives a new λ message from one of its children, then if x_i is not already instantiated,

```
BEGIN
    1. Compute the new value of $l(x_i)$ using operative formula 3;
    2. Compute the new value of $BEL(x_i = a_i)$ using operative formula 5;
    3. Post new $\lambda$ message to all $x_i$'s parents using operative formula 1;
    4. Post new $\Pi$ message to all $x_i$'s other children using operative formula 5;
END.
```

C. If a variable x_i receives a Π message from a parent, then

```
BEGIN
    If $x_i$ is not instantiated, then
        BEGIN
        1. Compute the new value of $\Pi(x_i)$ using operative formula 4;
        2. Compute the new value of $BEL(x_i = a_i)$ using operative formula 5;
        3. Post new $\Pi$ message to all $x_i$'s children using operative formula 2;
        END;
    If $l(x_i) \neq (1,1\ldots1)$, then
        Post new $\lambda$ message to all $x_i$'s  other parents using operative formula 1;
END.
```

3.2.5 Limitations of Bayesian Belief Networks

Despite the remarkable power and potential to address inferential processes, there are some inherent limitations and liabilities to Bayesian Belief networks.

The first problem is it is not good at representing and reasoning over actions and their interaction with observations. The reason is, in principle, actions are not part of standard probability theory. Probabilities capture and represent the casual relationship among observations while actions thought as the force to perturb these relationships [49]. Thus, the BBN, originally defined as a knowledge repository, expressed in the join probability distribution of random variables in a special domain, is unable to propagate the effect of an action.

The second problem is that for a particular BBN, causal relationships among random events are hard coded into the network [36]. Causal structure is pre-known and predefined without changes after the system is implemented. In the graph, the element in a node's parent set is fixed. If a causal relationship has to be updated after system is set up, the inference is crippled. BBN does not have this adjustment capability.

The third problem centers on the quality and extent of the prior beliefs used in Bayesian inference processing. A Bayesian network is only as useful as this prior knowledge is reliable. Either an excessively optimistic or pessimistic expectation of the quality of these prior beliefs will distort the entire network and invalidate the results. Related to this concern is the selection of the statistical distribution induced in modeling the data. Selecting the proper distribution model to describe the data has a notable effect on the quality of the resulting network.

In next chapter we introduce the causal network model, extended Bayesian Belief Network (eBBN). It extends BBN syntactically and semantically so that it has the ability of representation and reasoning over actions. This model is also capable of representing the changing casual structure.

3.3 Summary

In this chapter, we discuss the uncertainty and its impact on system performance. Generally speaking, uncertainty damages the supply chain management goals of an individual functional agent as well as MASCM performance. Since there is no way to block all uncertainty sources, one critical task of MASCM design and implementation is to formally study functional agent interaction in uncertain settings and analyze the relationship between uncertainty management mechanism and system performance. There is no existing work in this field. As we try to extend Bayesian Belief Networks to model functional agent interactions in uncertain environment, in the chapter, a summary of BBN research is also given.

Chapter 4 An Extended Bayesian Belief Network Model for MASCM

A MASCM defined in the Chapter 2 comprises a number of functional and informational agents. In this chapter, we focus our study on functional agents interaction in uncertain environment. Information agents are treat as parts of the marketplace. Without otherwise explanation, in this chapter, the term “agent” is equal to “functional agent.”

In a MASCM, business activities of an individual agent are defined as Order Fulfillment Processes (OFP), which are the efforts for the agent to manage individual supply chain. When an end order arrives in the system, a Virtual Supply Chain (VSC), consisting of agents from raw material suppliers to end customer agents, might emerge through multiple interconnected OFP practices. At the system level, the supply chain management is the combination of all agents’ activities ignited by one end order. The ultimate goal of the system management is to satisfy end customers’ requirements.

In the real life agents are exposed in an uncertain and dynamic environment. Their performances are affected by many unexpected physical failures such as electricity outrage, virus attacks and so on. In addition, agents must deal with sudden changes made by their trading partners. In either case, the agents have to adjust their behaviors accordingly. The uncertain events occur in a high frequency not only because of the complexity of current computer system but also because, in the electronic world, the business relationship is easy to setup at the low operational costs and there is little chance to punish agents for their strategic cheating behaviors.

Uncertainty damages the system performance. The dynamic environment creates more difficulties in performance improvement. To protect their common interests and improve social benefits, agents are inclined to cooperate with each other. Information flows can integrate agents' actions. Through information sharing and analysis, the negative impact of uncertainty on system performance can decrease. Therefore, from system design and implementation's perspective, studying how to establish an efficient and reasonable mechanism for agents' cooperation on dealing with uncertainty is critical. In this chapter, we present the research effort to develop a theoretical model that formalizes agents' interactions regarding to uncertainty in the environment. As the basis, the model can be further used to algorithm design for agents' uncertainty management and serve as the platform to analyze their relationship with various measurement of system performance.

The rest of this chapter is organized as follows: firstly, we describe observations on agent behaviors in OFPs; secondly, we give the formal description of MASCM states and introduce a simplified type of MASCM, called $MASCM_1$; thirdly, we give a general discussion on extended Bayesian Belief Network (eBBN) approach for $MASCM_1$ modeling; lastly, we present three related eBBN models in the increasing order of complexities.

4.1 Observations of agent interactions in OFP

In this section, we discuss several important observations on how agents operate their supply chains. These observations directly lead to our efforts on using causal network to study agent interactions in uncertain environment. They are listed below.

Firstly, in OFPs, agents interact with each other in order to reach mutual agreement and if they reach one, agents will keep contacting the other parties till it is finally resolved. In other words, a commitment plays a central role in agent interactions. However, agents' limited knowledge may cause a commitment to be created but fail to finish. When unexpected events occur, agents may abort the commitment. Therefore, the possibility of a commitment to accomplish successfully (unsuccessfully) can be used as a measurement for agents to estimate the impact of uncertainty. Accordingly, the uncertainty in the system can be represented as end customer agent's computation on the likelihood of commitment processing status. In this dissertation, we use the commitment failure probability (or agent's belief on commitment failure) to describe this likelihood.

In addition, OFPs describe supply and demand relationship between suppliers and their direct customer agent. The customer agent tries to reach a deal with direct suppliers. A supplier determines whether the product can be really delivered. If there is no agreement being reached or direct supplier withdraws the commitment, the customer agent may be forced to cancel its commitment on-hold. The common example is one of the key suppliers fails to fulfill its promise, e.g. failure of the CPU supplier to deliver the product on time can cause the PC factory to fail its agreement with its customers even though it has obtained all other components. That is, supply-demand relationship is one type of causal relationship since direct supplier's performance impacts customer agent's performance. Whether the demand of a customer agent can be satisfied or not depends on whether all its commitments made with direct suppliers can be completed. If they all accomplish, the customer agent's commitment finish successfully. Considering the disturbance of unexpected events during OFP period, it can be said that failure probability of commitments made by a pair of supply-customer agents are causally linked. The failure probability of a commitment made by direct supplier to the customer agent affects the one made by this customer agent to its downstream direct customer. Similarly, when further exploring the upstream of a supply chain, we can see current supplier agent's performance also depends on its own suppliers' performance. In other words, the casual relationship between commitment failure probabilities exists in any supply-demand connections. Therefore, the supply-demand relationship exhibited in OFPs triggered by one end order or in a VSC can be described as a casual chain (network) consisting of commitments with their failure probabilities.

Since each agent has certain level safety stock for each product it needs and can have alternative suppliers, one important property of the causal chain is that commitment failure from one of agent's direct suppliers might not eventually cause the commitment held in this agent to fail. In other words, the relationship between linked commitment failure probabilities cannot be defined as a deterministic function. It also has to be noticed that the causal chain represents two types of supply-demand relationship. One is supplier and customer relationship, which is stored as the static knowledge before agents start OFPs. The other one is buy-sell connections as we defined in Chapter 2, which is dynamically created and accompanied with the construction of an evolving VSC.

Lastly, agents' limited knowledge, internal function failure, and strategic actions are the uncertain sources that change the failure probability of a commitment. Their impact can be propagated through the whole VSC through interconnected OFPs and updates causally connected failure probabilities of commitments. Many fundamental procedures related to OFP can be illustrated through agent's update and propagation of commitment failure probabilities. For example, an agent's chain performance monitoring procedure can be described by its update of commitment failure probability based on the propagated information from up or down stream. Bilateral negotiation process between a supplier and direct customer agent also can be explained as the process that both sides work toward a mutually acceptable failure probability of a commitment, that is, searching for the low risk of a commitment accomplishment.

From above discussion we can see that agent interaction in an uncertain environment can be formalized as probability distribution propagation and update through causal links connected by agents' commitments made in OFPs.

4.2 System State Description

The states of a MASCM can be described by demand-supply relationships among agents. In this section, we study how to represent these relationships through directed graphs. We define the Supply-Demand Graph (SDG) and Dynamic Supply-Demand Graph (DSDG) to describe supplier-customer relationship and sell-buy connection respectively. We first give definitions of basic terms and notations.

We use symbol A_i to denote a functional agent. Accordingly, the MASCM is defined as an agent set S, $S = \{A_1, A_2, ..., A_i, ...A_n\}$. Based on the discussion in Chapter 2, each agent only provides one certain type of product; correspondingly, the total products provided by a MASCM can be denoted as set G, $G = \{g_1, g_2, ..., g_k, ...g_m\}$. The final product that sells to end customers is denoted as g_F and usually is the first element in G, that is, $g_1 = g_F$. In the system, two agents A_i and A_j can provide the same type of product that satisfies one direct customer needs. Therefore, if set S is finite, we have $m \leq n$. We use notation "G(A_i)" to express the relationship between the product and agent. For example, $g_k = G(A_i)$ means agent A_i can provide product or service g_k to its direct customers.

We use symbols A_i^s, A_i^c to denote A_i's direct supplier and customer agent sets, A_i^u and A_i^d to denote its upstream and downstream network respectively; $|A_i^s|$, $|A_i^c|$, $|A_i^u|$ and $|A_i^d|$ are used to denote the number of agents in each of these sets. Obviously, we have $A_i^s \subseteq A_i^u$ and $A_i^c \subseteq A_i^d$.

Symbol $B_i = \{b_i^1, b_i^2, ...b_i^k\}$ is an ordered and finite set of outstanding commitments that agent A_i currently hold. These commitments are made between agent A_i and its customer agents or itself (if it is an end customer agent). Symbol b_i^k denotes kth element in the set. Symbol $|B_i|$ represents the number of commitment in the set. We use symbol $B = \bigcup_i B_i$ to denote all commitments in the system. Symbol $A_i \cdot A_j$ is used as a label to describe the on-going business activities between agent A_i and its direct supplier agent A_j when they have a sell-buy connection. More specifically, in the dissertation, business activities refer to negotiation and commitment information exchange between a pair of direct supplier-customer agents. $A_i \cdot A_j$ is initialized (setup) by an order from agent A_i in order to complete the unsolved commitment it has; it is removed if agent A_i determines supplier A_j is no longer useful for its unsolved commitment, e.g., agent A_j withdraws its commitment to agent A_i. When system is reset, label of $A_i \cdot A_j$ is removed. In other words, whenever symbol $A_i \cdot A_j$ is used, it means there is a sell-buy connection between two agents; they are either doing business or the supplier has successfully finished the commitment. At this time we also say agent A_i (A_j) is active.

4.2.1 Supply-Demand Graph (SDG)

The Supply-Demand Graph (SDG) describes supplier customer relationship among agents. Using these symbols, we give the definition of SDG as follows.

Definition 4.1. SDG=<V, E>, V=S, E={$< A_i, A_j >$| $A_i \in$ S, $A_j \in$ A_i^s}. Symbol SDG(S) denotes the SDG corresponding to a specific agent system S.

From the definition, we know SDG is a directed graph that describes the long-term knowledge on supply-demand relationship among agents. It is used to represent initial state of a MASCM, in which no end orders arrive to the system and no commitments as been made by any agents. Usually, each system has one SDG.

4.2.2 Dynamic Supply-Demand Graph (DSDG)

Dynamic Supply-Demand Graph (DSDG) represents sell-buy connections dynamically established by OFPs after an end order comes to the system. We give the formal definition of VSC and Dynamic Supply-Demand Graph (DSDG) as follows,

Definition 4.2. $VSC_k = \{ A_i,\ A_j \mid \exists A_i \cdot A_j,\ A_i \in \mathrm{S},\ A_j \in A_i^s;\ A_i,\ A_j \in A_k^u,\ \mathrm{G}(A_k) = g_F \}$.

This definition covers both two types of chain, evolving VSC and completed VSC. At any given time, we use symbol VSC(S) to identify the set of virtual supply chains, VSC_k, currently in the system.

Definition 4.3. DSDG=<VSC(S), L>, L= $\{ < A_i, A_j > \mid \exists A_i \cdot A_j,\ A_i \in \mathrm{S},\ A_j \in A_i^s \}$. Symbol DSDG(S) denotes the DSDG corresponds a specific agent system S.

From the definition we know the sell-buy connections or OFPs can be described by DSDG. In the lifetime of a MASCM, at any given time, one of its states that contain actual business activities is described by a DSDG. In other words, the MASCM runtime states are determined by a set of DSDGs.

4.3 $MASCM_1$

A MASCM is a complex system. To simplify the study, in this section, we introduce assumptions posted on the MASCM settings. Based on these assumptions, we define $MASCM_1$, which serves as the basic framework for our future discussion.

4.3.1 Assumptions

To define a $MASCM_1$, we make followings assumptions on system settings.

Assumption 4.1. There is only one end customer agent in the system. It is denoted as $A_1 \in$ S and $A_1^c = \phi$.

Assumption 4.2. $\forall$ $A_i \in$ S, if $i \neq 1$, then $| A_i^c | = 1$. That is, each agent, except agent A_1, has exactly one customer.

Assumption 4.1 and Assumption 4.2 simplify system architecture.

We have following assumptions on agent transactions.

Assumption 4.3. At any given time, $|B_i|\leq 1$. That is, each agent make/hold no more than one commitment to its customer agent at a time.

Assumption 4.4. At any given time, if $\exists\ A_i \cdot A_j$, $\exists\ A_i \cdot A_k$, $G(A_j) = G(A_k)$, then $j = k$. That is, no agent will order the same product from two or more different suppliers at the same time.

Following assumptions are for agent internal business logics.

Assumption 4.5. (Accountability Assumption)

$\forall\ A_i \in$ S, the commitment set B_i has certain probability to fail when any of its demand for certain service is not satisfied. However, if all these demands are satisfied, the commitment set B_i is resolved successfully.

Assumption 4.5 says any failure from an agent A_i's direct supplier in finishing commitments may cause its own commitment to fail. When all commitments (if there are any) have been resolved by its direct suppliers, the commitment that an agent made to its own customer agent is considered as successful accomplished. However, as we discussed in Section 4.1, a single supplier failure might not necessarily cause a failure in customer agent to definitely happen. It is the reason that we use the term of "certain probability to fail," instead of "fail," to describe the situation.

Assumption 4.6. (Process Independence)

Different OFPs triggered by A_i ($A_i \in$ S) are independent to each other.

Assumption 4.6 regulates that OFPs between two agents, agent A_i and one of its direct supplier agent A_j, are not created or affected by other on-going or finished OFPs initiated by A_i. Process Independence can derive the *Exception Independence* [41], which indicates the influence from a direct supplier that causes the customer agent fails in commitment processing is independent to one from another supplier. It is an important assumption when we analyze the run-time uncertainty of a system.

The MASCM that follows above assumptions (Assumption 4.1 - Assumption 4.6) is the system we intend to model and study in the dissertation. This type of MASCM can be defined as follows,

Definition 4.4. Symbol $MASCM_1$ represents an agent system S that satisfies Assumption 4.1 to Assumption 4.6.

In the future, unless specified otherwise any agent system S represents a $MASCM_1$.

4.3.2 Properties of $MASCM_1$

From the assumptions that define $MASCM_1$, we can directly derive some useful properties for our study. For example from Assumption 4.3, we know that an agent at most hold one commitment to its customer. Thus, we can use B_i to describe the unsolved commitment agent A_i hold instead of b_i^1. Specifically, the commitment hold by end customer agent A_1 is B_1. Since a VSC is triggered by an order, based on Assumption 4.1 through Assumption 4.4, it is obvious that at any given time, there is only one VSC in the system. Therefore, instead of using "VSC_k," we use symbol of "VSC" to represent the unique chain in the system.

Proposition 4.1. SDG(S) is a tree rooted as A_1.

Proof. From Assumption4.1 and Assumption 4.2, we know any nodes in the graph has at most one parent node and node A_1 does not have any parents. Thus, SDG is a tree rooted as A_1.

□

Corollary 4.1. $\forall A_i, A_j \in A_k^s$ $(i \neq j)$, $A_i^u \cap A_j^u = \phi$

Proof. It can be proved from Proposition 4.1.

□

From Chapter 2 we know that the concept of tier in supply chain management research is based on the relative distance to the concerned customer [4][34], i.e. a direct supplier is one tier close to its customer than its direct suppliers are. In some situations, this definition is ambiguous. For example, if agent A_k is the supplier for both agent A_i and its direct supplier agent A_j, we cannot determine the relative distance between A_i and A_k. However, based on Corollary 4.1, in $MASCM_1$, there are no agents that can be another agent's direct and indirect suppliers at the same time. Therefore, these situations would not occur in the system we study. The following rules is used to determine the exact tier an agent sits in.

Definition 4.5. The rules blow are used to determine the tier an agent belongs to,

i. t(A_1)=0;.

ii. t(A_j)=t(A_i)+1; if $A_j \in A_i^s$.

Definition 4.5 gives a measurement of the distance from an agent to an end customer agent or the height of from a leave node to root note in the SDG.

For transactions among agents that are at the same tier, we have following proposition,

Proposition 4.2. If $\exists\, A_i \cdot A_j$, then t(A_j)≠t(A_i).

Proof. From definition of symbol $A_i \cdot A_j$, we know if $\exists\, A_i \cdot A_j$, then $A_j \in A_i^s$. However, the Definition 4.5 tells us at this time point, t(A_j)=t(A_i)+1. □

From Proposition 4.2, we can see in our setting, at any time, there are no business activities between the agents at the same tier.

For the graph structure we have following proposition.

Proposition 4.3. DSDG(S) is a tree.

Proof. Obviously, we have L⊆E, VSC(S)⊆S. That is, any time graph DSDG is a sub-graph of graph SDG. Graph SDG(S) is a tree, so is graph DSDG(S).

□

Proposition 4.3 shows both sell-buy connections among agents and system final solution to an end order can be organized as a tree.

From the discussion above we can see that the states of $MASCM_1$ are determined by supply-demand relationship among agents. This relationship can be described by a SDG (static) and a group of DSDG (run-time). Since the demand-supply relationship is a special type of causal relationship as we discussed in section 4.1.1, the agent interactions in a $MASCM_1$ can be modeled using a causal network. In other words, given the SDG(S) and the set of DSDG(S) for certain $MASCM_1$, we can define a corresponding belief network.

4.4 Modeling agent interactions in uncertain environments

In this section we generally discuss how to establish a Bayesian Belief Network (BBN) model to formalize agent interactions in MASCM under uncertain settings. We first define the basic network components and explain their physical meanings. Then, we give a brief discussion on probability propagation and uncertainty analysis.

4.4.1 Modeling commitment failure probability as an agents' belief

In section 4.3.1, we know at any given time, each agent only holds one commitment (to its customer agent or itself). It leads us to use one single random variable to describe the situation of commitment processing. We give the definition of the random variable below,

Definition 4.6. Commitment failure variable x_i is a binary random variable. It represents the current processing situation of the commitment B_i made by agent A_i to its customer; $x_i = 1$ means the commitment B_i fails; $x_i = 0$ means the commitment B_i is successfully accomplished.

In the BBN, variable x_i is represented as a node. In the future, without further explanation, the node and random variable are interchangeable terms in the context of BBN.

As a random variable, each x_i has a probability distribution, which is denoted as $p(x_i)$. From the discussion in section 4.1, this distribution and its update describe an agent's observation on the impact of uncertainty, that is, in the term of BBN, an agent's *belief* on how likely a commitment may fail.

Based on Definition 4.6, agent A_i's initial belief on commitment failure can be represented as $p(x_i = 1)$. It reflexes agent A_i's estimation to complete the most common case of orders from its direct customer agent under the current situation e.g. inventories of each material and how likely the products the agent requests can be obtained based on the past experiences. Since its belief will be updated when a commitment fails (resolves successfully), or other agents change beliefs, agent A_i's belief on commitment failure in fact is a posterior probability of x_i, written as $p(x_i = 1 | E)$. It represents the overall likelihood of proposition $x_i = 1$ considering all evidence system so far has received, where E represents all facts in the system or the value combination of all instantiated commitment failure variables in the corresponding causal network. E can be an empty set. In this case $p(x_i = 1 | E) = p(x_i = 1)$.

The way we define agent's belief is identical to the one given by Judea Peal in his book [41]. The concept of "belief" in [41] represents the dynamic value of a node's probability in a knowledge base. Followed the convention in [41] $p(x_i = 1 | E)$ can be written as $BEL(x_i = 1)$. In the dissertation these two expressions are equal.

System uncertainty can be defined as the combination of agents' beliefs in a VSC. These agents are currently involved in OFPs in order to resolve commitment B_1 held by end customer agent. Following Judea Peal's convention, we have the following definition.

Definition 4.7. At any given time, the uncertainty in the system S is $UNC(S) = BEL(x_1 = 1)$.

Directly from Definition 4.7, we know if the end order finishes successfully, $UNC(S)$=0; when it fails, $UNC(S)$=1; otherwise, $0 \le UNC(S) \le 1$. Or we can say, when an end order is successfully solved, the impacts of uncertainty observed by all agents are eliminated.

4.4.2 Modeling causal relationship as directed links

Agents' beliefs represent observation on the impacts of uncertainty. They are the pivots to formalize agents' interaction in uncertain environment. In this section, we study how to use and extend the concept of "link" of BBN to represent the causal relationship related to agents' beliefs.

From the discussion in 4.1, we can see for a given commitment, two types of sources can affect its failure probability. One exhibits in the supplier-demand relationship. That is, the direct suppliers' commitment failure probability changes cause this commitment failure probability to change accordingly. Other sources are internally from an agent. They can be internal function failures or strategic actions. In other words, in a system, direct suppliers' beliefs, and specified or unspecified internal situation changes affect an agent's belief. Following definitions in BBN, a directed arc, called *link*, the representation of relationship from the cause to the effect, is drawn from node of commitment failure variable associated with a supplier to the one associated with its customer. Similarly, there should be links from all possible internal unexpected changes to a commitment failure variable. In the realities, it is impossible and unnecessary to enumerate internal causes that can change an agent's belief. An agent is a proactive entity pursues business goal on half of its owners. Thus, an agent's strategic actions (decisions) are considered as important source that change its belief. In this dissertation, we study the causal relationship between an agent's belief and its actions related to supplier selection, commitment cancellation. For other unspecified internal sources, their impact on agent's belief change (distribution of commitment failure variable change) is recognized even though they may not be specified.

There are two difficulties when applying the concept of link to causal relationship related to agent beliefs.

First, although there should be a link between two commitment failure variables associated with a pair of supply and customer agents based on their long-term supply-demand relationship this link might not correctly catch the actual the sell-buy connection between them during a specific end order solving time period. For example, if agent A_j is one of agent A_i's direct suppliers, the impact of A_j's belief on the one of A_i can be represented as the link between x_j and x_i. However, let us consider the following scenario: between the time point when an end order arrives at the system and the one when it is finally resolved, no transactions are between A_i and A_k. In this situation, we cannot claim beliefs about commitment failure associated with agent A_i and A_k are casually related. The situation occurs whenever the direct supplier agent A_j does not have a sell-buy connection agent A_i at runtime. This problem is directly related to the representation and inference capability limitation of traditional BBN. Many researches found that BBN lacks the power to represent and infer system probability when causal structure can dynamically change or update (see Chapter 3). We will further address this problem and present our solution in Section 4.6 when we introduce model of $eBBN_1$.

Second, since actions are not compatible concepts with traditional BBN definition, when studying their effects to agent belief, we have to cautiously define their intervention to the network, and solve ramification and concurrent problems. In addition, we also have to extend the concept of link itself so we can have a proper representation between the cause of action and the effect of an agent's belief. We will further address this problem and present our solution in Section 4.7 when we introduce model of $eBBN_2$.

One thing we have to notice that in order to properly represent causal relationship, except the commitment failure variable, we define some new type of nodes into the network. For instance, a type of dummy nodes is introduced into the BBN to represent effect of decision consequence. Nodes that represent agent actions also have to be defined in the model. In addition, the concept of link needs to capture both short-term sell-buy connections and long-term supplier and customer relationship. Therefore, links are not necessarily between two commitment failure variables. We give the definition of link used in this dissertation as follows.

Definition 4.8. A causal link is a direct arc connected two nodes, from the causal to the effect.

Usually, we use symbol <causal_node, effect_node> to represent a link literately (causal_node and effect_node represent two random variables defined in the model. They can be regular nodes, dummy nodes and action nodes defined in Section 4.7.). For example, if the commitment failure variable x_j is the direct cause of node x_i, the explicit link between them is represented as $< x_j, x_i >$. It is depicted as the following figure,

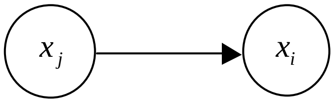


Figure 4.1 A link in BBN for MASCM

4.4.3 Modeling information sharing in uncertain environment

The uncertain information that an agent is willing to share with others should be the one that do not expose its internal sensitive data. An agent's belief hides a lot of details related to its internal situations. It can be used to describe the type of information agents can send to its business parties. Moreover, any incoming information may change agent business behaviors. Thus, when using nodes to represent the observed impact of uncertainty and casual links to model the relationship of these impacts, agents' information sharing under uncertain settings can be easily described as belief update and propagation follow certain rules dependent on chain architectures.

The theoretical model, including nodes, links, related conditional probability distribution, and rules on nodes' probability distribution update and propagation, formalize agents' cooperative efforts to decrease the negative impact of uncertainty on chain management. This model describes agents' interactions in each of MASCM states defined by SDG and DSDG in uncertain environment, and is called extended Bayesian Belief Network (eBBN) as it has the ability to represent and infer on dynamic causal structure and the effect of actions.

To establish an eBBN model for a MASCM, we need to determine graph structure and probability distribution for each node at any given time. In the context of this dissertation, the former topic is called as *syntax analysis* of the model and latter one is called *semantics analysis*.

In the rest of this chapter, we define eBBN models for $MASCM_1$. We first study the model for a special type of $MASCM_1$, called $MASCM_0$. In $MASCM_0$ each agent only has one direct supplier to provide one type of goods it needs, and all of agents in the system are active at any time. The eBBN model of $MASCM_0$ is called $eBBN_0$. Then, we discuss a more complex eBBN model, called $eBBN_1$, for $MASCM_1$, which can represent dynamic causal structures. Agents' action are not directly represented by both model of $eBBN_0$ and $eBBN_1$. Lastly, we introduce the concept of action into $eBBN_1$ leading to model of $eBBN_2$. Through model analysis we have found that both $eBBN_1$ and $eBBN_2$ have similar syntax and semantics as model $eBBN_0$ does.

4.5 The model of $eBBN_0$

In this section, we study an eBBN model for a simply type $MASCM_1$, called $MASCM_0$, in which each agent has to order certain product fom one and only one supplier and all agents are involved in OFPs triggered by an end order at any time. This model is called $eBBN_0$. Agent actions are not explicitly represented in the model. Model of $eBBN_0$ serves as the basis of construction and analysis for more complex types of models. In this section, first, we define model of $eBBN_0$, and then we represent syntax and semantic analysis for $eBBN_0$.

4.5.1 The basic definitions of $eBBN_0$

System $MASCM_0$ is defined as follows,

Definition 4.9. $MASCM_0$ is $MASCM_1$, $\forall A_i \in$ S, if A_j, $A_k \in A_i^s$, $g_l = G(A_j)$, $g_m = G(A_k)$, then $g_l \neq g_m$; VSC(S)=S.

In system $MASCM_0$, an agent has implicitly chosen all direct suppliers to satisfy its demands. At any time, we have SDG(S)=DSDG(S). An agent's commitment processing is affected by its direct suppliers' performance. That is, an agent's belief on commitment failure is influenced by ones from its direct suppliers and affects direct customer's belief.

The model of $eBBN_0$ is defined as follows,

Definition 4.10. $eBBN_0$=(V_0, EL_0), V_0={x_i | x_i associates with A_i, $A_i \in$ S}, EL_0={< x_j, x_i >| $\exists A_i \cdot A_j$, A_j, $A_i \in$ S}.

In eBBN model for $MASCM_1$, we use p_i to denote parent set of node x_i. In $eBBN_0$, p_i={x_j| $\exists$< x_j, x_i >}. The conditional probability distribution table of node x_i can be recorded as $p(x_i = 1 | p_i)$.

4.5.2 Syntax and semantic analysis of model $eBBN_0$

In this section, we analyze the syntax and semantics of $eBBN_0$. For the syntax of $eBBN_0$, we have following theorem.

Theorem 4.1. Model of $eBBN_0$ is a reverse tree.

Proof. Node $x_i \in$ $eBBN_0$ if and only if $\exists A_i \in$ S; Link <x_j, x_i> $\in$ $eBBN_0$ if and only if $\exists A_i \cdot A_j$. From Definition 4.4, we know if $\exists A_i \cdot A_j$, A_j $\in$ A_i^s, we have < A_i, A_j > in DSDG. Thus, $eBBN_0$ is identical to DSDG(S) at any given time, except the name of nodes and reversed links. From Proposition 4.3, we know DSDG(S) is a tree. Then we know $eBBN_0$ is a reverse tree.

□

Now, we prove that model $eBBN_0$ is semantically equivalent to Noisy-Or network. In order to proceed with our argument, we introduce the following random variable.

Definition 4.11. Let random variable c_{ji} denote the causal connection from a commitment failure variable x_j to x_i. If $c_{ji}=1$, then x_j=1 indeed causes x_i=1. Otherwise, x_j=1 does not affect x_i.

Node c_{ji} is used to describe the working status of the underlying causal mechanism between a pair of direct supplier and customer agents.

Lemma 4.1. The model of $eBBN_0$ is a Noisy-Or network.

Proof. Based on Assumption 4.5 (Accountability Assumption) and Assumption 4.6 (Process Independence or Exception Independence), in $MASCM_0$, at any given time, the business logic about the processing failure of commitments between agent A_i and its direct supplier set A_i^s can be described by the proposition logic among the commitment failure variables in $eBBN_0$ using the following equation,

$$(x_i = 1) \equiv \vee_j (x_j = 1 \wedge c_{ji} = 1)\ ,\ x_j \in p_i\ , \forall x_i \in eBBN_0\ . \tag{4.1}$$

Equation (4.1) is a standard way to describe the logic relationship among nodes in a Noisy-Or BBN [42]. Thus, lemma is proved.

□

From the Lemma 4.1, we can directly determine probability distribution of commitment failure variables in the $eBBN_0$ using the followed theorem.

Theorem 4.2. $BEL(x_i = 1) = 1 - \prod_{x_j \in p_i} (1 - e_{ji} BEL(x_j = 1))\ \ \forall x_i \in eBBN_0$.

From the Equation (4.1) we have,

$$p(x_i = 1 \mid E) = p(\vee_j (x_j = 1 \wedge c_{ji} = 1) \mid E)\ ,\ x_j \in p_i\ . \tag{4.2}$$

If we let $e_{ji} = p(c_{ji} = 1 \mid x_j = 1)$, equation (4.2) can be easily transformed into the following equation, (this detailed process can be found in [41])

$$p(x_i = 1 \mid E) = 1 - \prod_{x_j \in p_i} (1 - e_{ji} p(x_j = 1 \mid E))\ . \tag{4.3}$$

That is, we have,

$$BEL(x_i = 1) = 1 - \prod_{x_j \in p_i} (1 - e_{ji} BEL(x_j = 1))\ . \tag{4.4}$$

□

According to equation (4.4), at given time, the uncertainty in system S can be represented as follows,

$$UNC(S) = BEL(x_1 = 1) = 1 - \prod_{x_j \in p_1} (1 - e_{ji} BEL(x_j = 1)) \tag{4.5}$$

Equation (4.5) shows we can use model of $eBBN_0$ to determine uncertainty in the system.

The conditional probability e_{ji}, defined in Theorem 4.2 as $e_{ji} = p(c_{ji} = 1 | x_j = 1)$, is called the *link strength* of link <x_j, x_i>. Generally speaking, the link strength represents the causal strength between two random variables (events). For example, e_{ji} is used to measure the impact of the agent A_j's ($A_j \in A_i^s$) belief on commitment failure to the one of agent A_i. The value of e_{ji} is a real number between 0 and 1. When e_{ji} is equal to 1, the commitment failure from A_j will independently cause the commitment in A_i to fail. The value of e_{ji} encodes agent A_i knowledge on interaction with agent A_j regarding to uncertainty in the system. As an entry of CPD associated of x_i, e_{ji} is pre-stored and can be adjusted by agent A_i.

Value of e_{ji} is dependent on many business factors that agent A_i concerns, e.g. the supplier reputation and the importance of a product to current commitment. The higher the link strength is, the larger the impact of one commitment failure on the other one is.

To large extents, the value of e_{ji} represents the importance of the product, which supplier agent A_j promises to deliver, on the accomplishment of the commitment that agent A_i currently holds. For example, a laptop assembling factory has high "safety stock" on all components except CPU. Then, the accomplishment of the commitment, which is made with the retailer agent, who serves as the agent's direct customer, depends more on commitments made by CPU providers than other suppliers. In this case, the higher the value e_{ji} is, the more important the products that supplier agent A_j sell is.

Given a group of supplier agents, e.g. A_j and A_k, that can provide the same product to agent A_i, the value of e_{ji} and e_{ki} also reflects their business reputation evluated by agent A_i. This type of knowledge can be generated by agent A_i's past business experience with A_j and A_k or obtained directly by sending queries to informational agents. A direct supplier with better reputation has less value of the link strength. For example, if agent A_i concludes supplier A_j has better reputation than A_k, the value e_{ji} is less than e_{ki}.

The value of e_{ji} is also affected by time. For example, in an OFP, agent A_i has made two commitments with supplier agent A_j. One is reached at initial time when no other suppliers ever made or finished commitments with A_i. The other is created later when all other suppliers has been finished their commitments with A_i. Then, at the second time point, the value of e_{ji} should be higher than the early time since at that time, commitment failure in A_j can directly cause the commitment held by A_i to fail.

Usually, link strength can be represented as a function $e_{ji} = f_{e_{ji}}(p, r, t)$ (p is the product importance to current commitment, r represent supplier reputation, t is time.). As we discussed above, when it is closer to the finish time, the current commitment held by customer is more sensitive to the commitment made by direct suppliers. That is, using the notation above, we have, $e_{ji} = f_{e_{ji}}(p, r, t) \to 1$, when $t \to t_0$ (t_0 is the deadline of the current commitment in A_i). However, the detailed implementation of this function is decided by the owner of the agent A_i and is different from agents to agents.

4.5.3 Further discussion

In $MASCM_0$, all agents are involved in OFPs triggered by an incoming end order. Supplier-customer relationships imply sell-buy connections and vice versa. That is, at any time, DSDG is equivalent to SDG. Therefore, DSDG(S) and SDG(S) are directly mapped into model $eBBN_0$, which represents both static and run-time casual relationship exhibited in a system. This is an important attribute that distinguishes $eBBN_0$ from other models introduced later.

4.6 The model of $eBBN_1$

In this section, we study an eBBN model, called $eBBN_1$, for $MASCM_1$. Agents' actions are not represented directly in this model. Since $MASCM_0$ is a special case of $MASCM_1$, mode $eBBN_1$ is built based on $eBBN_0$. We also show they have similar equations to compute probability distribution of commitment failure variables.

4.6.1 The definitions of $eBBN_1$

One significant difference between $MASCM_0$s from other $MASCM_1$s is that in the former ones, for a certain product, a customer agent only has one direct supplier agent. Another difference is in a $MASCM_1$, not all agents are involved in the chain activities. Thus, when modeling $MASCM_1$ as $eBBN_1$, it is necessary to model the consequence of agents' decision on supplier (provider) selection and consider their impacts. That is, $eBBN_1$ needs to model dynamically changing casual structures. To capture these changes and at the same time keep model representation consistent, we introduce two types of nodes. One represents the consequence of agent decisions. Another is used to represent the impact of this consequence on supply-demand relationship. The definitions of these nodes are as follows.

Definition 4.12. Link selection variable l_{ji} is a binary random variable. If l_{ji}=1, agent A_i is dealing with supplier agent A_j actively. l_{ji}=0, agent A_j does not currently get involved in agent A_i's OFPs.

Node l_{ji} is associated with A_i and represents an observable consequence of agent A_i's decisions. These decisions are related to negotiation partner selection or switching. In the model, for each pair of commitment failure variables associated with a customer agent and one of its direct supplier, there is a link selection variable. In $MASCM_1$, each agent only chooses one direct supplier for certain service it needs at a time. Thus, we have following lemma.

Lemma 4.2. Initially, l_{ji}=0, $l_{ji} \in eBBN_1$. At any given time, if l_{ji}=1 and l_{ki}=1, A_j, $A_k \in A_i^s$, G(A_j)=G(A_k), then $j = k$.

Proof. The proof is directly from Definition 4.12 and Assumption 4.4.

□

The node y_{ji} that describes impact of decisions on casual relationship between commitment failure variables associated with a pair of direct connected supplier-customer agents. The definition is as follows.

Definition 4.13. The binary random variable y_{ji} has two parents, l_{ji} and x_j, and one child x_i. Its conditional probability distribution is below.

$p(y_{ji} = x_j \mid l_{ji} = 1, x_j) = p(y_{ji} = x_j \mid l_{ji} = 1) = 1;$

$p(y_{ji} = 0 \mid l_{ji} = 0, x_j) = p(y_{ji} = 0 \mid l_{ji} = 0) = 1;$

From the Definition 4.13 we can see node y_{ji} associated with agent A_i and represents its observation on whether there sell-buy connection between agent A_j and A_i. It also describes how agents' beliefs interact with each other if there indeed exists a connection. This observation is one transaction (single OFP) based. When l_{ji}=1, both agent A_j and agent A_i are active ($\exists$ $A_i \cdot A_j$). Agent A_j's belief on commitment failure affects the one associated with agent A_i. Commitment failure variables x_i and x_j are causally related; when l_{ji}=0, agent A_j is not active and agents' beliefs are not causally related.

From the discussion above, we can say that, to certain degree, the dummy node y_{ji} is a "gate" between two commitment failure variables and is controlled by node l_{ji}. From the viewpoint of causal node x_j, when the gate is open, node y_{ji} takes its effect and become its proxy node; from the perspective of effect node x_i, node y_{ji} is the node that causes its probability to update. They should be set between two commitment failure variables associated with a direct connected supply-demand agents. In the model, for a pair of commitment failure variables associated with supplier-customre agents, e.g. node x_i and x_j, we add links $<x_j, y_{ji}>$, $<y_{ji}, x_i>$ and $<l_{ji}, y_{ji}>$.

Similar to c_{ji}, we use c'_{ji} to denote the causal connection from a commitment failure variable y_{ji} to x_i. If c'_{ji}=1, y_{ji}=1 indeed causes x_i=1. Otherwise, y_{ji}= 1 does not affect x_i. We have following definition,

Definition 4.14. $p(c'_{ji} = 1 | y_{ji} = 1) = e_{ji}$.

The relationship among random variables x_j, x_i, y_{ji} and l_{ji} is depicted by the following graph,

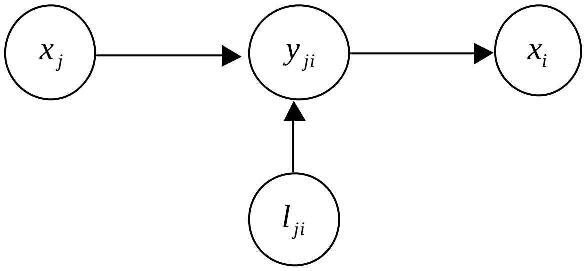


Figure 4.2 the relationship among random variables

By adding these new types of nodes and links into $eBBN_0$, the $eBBN_1$ model for a $MASCM_1$ is established. Its definition is given as follows,

Definition 4.15. $eBBN_1$=(V_1, EL_1), V_1={x_i, l_{ji}, y_{ji}| x_i, l_{ji} and y_{ji} associates with A_i, $A_i \in$ S}, EL_1={ $<x_j, y_{ji}>$, $<y_{ji}, x_i>$, $<l_{ji}, y_{ji}>$ | A_j, $A_i \in$ S }.

Obviously, in $eBBN_1$, p_i={ y_{ji}| $\exists$ $<y_{ji}, x_i>$}. We define other groups of nodes set in $eBBN_1$ as follows,

Definition 4.16. p_{ji}={x_j, l_{ji}| $\exists$<x_j, y_{ji}>, <l_{ji}, y_{ji}>}, Y_i^k={y_{ji}|$y_{ji} \in p_i$, $g_k = G(A_j)$}; p_i^k={x_j| $x_j \in p_{ji}$, $g_k = G(A_j)$}, L_i^k={l_{ji}| $\exists$ <l_{ji}, y_{ji}>, $y_{ji} \in Y_i^k$ }

Using the definition above, we have following proposition,

Proposition 4.4. If y_{ji}, $y_{ki} \in Y_i^k$, at a given time point, and y_{ji}≠0, y_{ki}≠0 then $k = i$. In other words, any given time at most one $y_{ji} \in Y_i^k$ takes non-zero value.

Proof. It is directly from Lemma 4.2 and Definition 4.13.

□

From Proposition 4.4 we know there is an Exclusive-OR relationship among the proposition y_{ji}≠0 ($y_{ji} \in Y_i^k$). We use the symbol $\oplus$ to denote Exclusive-Or relationship. Then we have the following proposition,

Proposition 4.5. At given time, y_{ki}, $y_{ji} \in Y_i^k$,

$p(c'_{ji} = 1 \mid \oplus y_{ki} \neq 0) = e_{ji}$, if y_{ji}=1;

Proof.

If y_{ji}=1, from Lemma 4.2. , we have y_{ki}=0 ($\forall$ $k \neq j$). That is,

$$p(c'_{ji} = 1 \mid \oplus y_{ki} \neq 0)$$

$$= p(c'_{ji} = 1 \mid y_{ji} = 1, \wedge_{k \neq j} y_{ki} = 0) \tag{4.6}$$

From the definition of c'_{ji} and Process Independence (Assumption 4.6), we have,

$$p(c'_{ji} = 1 \mid y_{ji} = 1, \wedge_{k \neq j} y_{ki} = 0) = p(c'_{ji} = 1 \mid y_{ji} = 1) = e_{ji}. \tag{4.7}$$

From Equation (4.7), this proposition is proved.

□

Conditional probability of $p(c'_{ji} = 1 \mid \oplus y_{ki} \neq 0)$ can be seen as the causal strength between beliefs in a direct supply agent group (produce the same type of product) and belief of their common direct customer agent A_i. It expresses, at the given time, the performance importance of certain agent group to fulfill the commitment B_i that agent A_i currently holds.

4.6.2 Syntax and semantic analysis of model $eBBN_1$

In this section we discuss the syntax and semantic of model $eBBN_1$.

Theorem 4.3. Mode of $eBBN_1$ is a reverse tree.

Proof. From Figure 4.2, we can see that after we introduce new type of nodes and links into $eBBN_0$, there is no cycle created among the nodes. Model of $eBBN_0$ is a reverse tree, so is $eBBN_1$.

□

Agent belief and system uncertainty can be determined by model $eBBN_1$ using following theorem.

Theorem 4.4. $p(x_i = 1 | E) = 1 - \prod_{l_{ji}=1}(1 - e_{ji} p(x_j = 1 | E))$, $A_i \in$ VSC(S).

Proof. For any agent $A_i \in$ VSC(S), we can categorize agent A_i's the direct supply set A_i^s into different groups, each of which provides one product that A_i needs to fulfill the commitment it made to the customer agent. Each agent group is written as A_i^k. If we treat A_i^k as a super direct supplier agent of agent A_i, a system consisting of ($A_i, A_i^1, \ldots, A_i^k, \ldots, A_i^l$) (k$\leq$l) (we assume agent A_i needs l different types of product to finish its own commitment to the customer) can be viewed as a $MASCM_0$.

Thus, for node x_i associate with agent $A_i \in$ VSC(S), we do the following reduction. We define pseudo-node X_q^k associates with agent group A_i^k, which contains the node set $L_i^k \cup Y_i^k \cup p_i^k$; if X_q^k=1, service g_k fails to deliver, if X_q^k=0, service g_k has no influences on current commitment that agent A_i holds. From here we can see that pseudo-node X_q^k and x_i are causal related. We draw a pseudo link directed from X_q^k to x_i. That is, random variable set $(x_i, X_j^1,..., X_j^k, \dots, X_j^l)$ and corresponding links defines an eBBN model for system ($A_i, A_i^1,..., A_i^k, \dots, A_i^l$). This model is $eBBN_0$. According to Assumption 4.5 and 4.6, we also know A_i independently evaluate the performance for each group. Therefore, if pseudo-link strength of link $<X_q^k, x_i>$ is defined as E_{qi}^k, based on Theorem 4.2, the following equation holds,

$$p(x_i = 1 | E) = 1 - \prod_{k=1}^{l} (1 - E_{qi}^k p(X_j^k = 1 | E)) \qquad (4.8)$$

From the Assumption 4.6, when agent A_j is active, we have following equation,

$$X_j^k\text{=1} \equiv (x_j = 1) \wedge (l_{ji} = 1) \wedge (y_{ji} = x_j),\ y_{ji} \in Y_i^k,\ x_j \in p_{ji}\ \ l_{ji} \in L_i^k \qquad (4.9)$$

Therefore,

$$p(X_j^k = 1 | E) = p(x_j = 1, l_{ji} = 1, y_{ji} = x_j | E)$$

$$p(X_j^k = 1 | E) = p(y_{ji} = x_j | x_j = 1, l_{ji} = 1)\ p(l_{ji} = 1)\ p(x_j = 1 | E) \qquad (4.10)$$

From Definition 4.13, Equation (4.10) can be written as following equation,

$$p(X_j^k = 1 | E) = p(x_j = 1 | E) \qquad (4.11)$$

The value of E_{qi}^k describes one group agent performance to the commitment failure that A_i holds to its customer. It semantically equals to $p(c'_{ji} = 1 | \oplus y_{ki} \neq 0)$ ($y_{ji} \in Y_i^k$). When A_j is active, $l_{ji} = 1$, according to Proposition (4.5), we have,

$$E_{qi}^k = p(c'_{ji} = 1 | \oplus y_{ki} \neq 0) = e_{ji} \qquad (4.12)$$

Thus, from Equation (4.11) and (4.12), Equation (4.8) can be rewritten as follows,

$$p(x_i = 1 | E) = 1 - \prod_{l_{ji}=1} (1 - e_{ji} p(x_j = 1 | E)) \qquad (4.13)$$

Or we have,

$$BEL(x_i = 1) = 1 - \prod_{l_{ji}=1} (1 - e_{ji} BEL(x_j = 1)) \qquad (4.14)$$

□

Similar to Equation (4.14), we have following equation to determine system uncertainty,

$$UNC(S) = BEL(x_1 = 1) = 1 - \prod_{l_{j1}=1} (1 - e_{j1} BEL(x_1 = 1)) \ .$$

4.6.3 Further discussion on $eBBN_1$

$eBBN_1$ model can represent and infer dynamic casual relationships through introduction of dummy node y_{ji} for each supplier agent A_j that affects agent A_i's belief. If supplier agent A_j is involved an OFP triggered by agent A_i, node y_{ji} becomes invisible between node x_j and x_i. Two agent beliefs are thus causally connected. However, at the time when agent A_j is not active, node y_{ji} blocks the uncertainty propagation between these two nodes. In this way dummy node y_{ji} represents the random event related to the consequence of agent's supplier selection on supply-deman relationship. This allows model of $eBBN_1$ is maximally compatible with probability theory, and allow agents to make reasoning over causal structure changes with uncertainty.

4.7 The model of $eBBN_2$

In previous sections, we discuss how to use eBBN to model a $MASCM_1$ without explicitly representing agents' actions. However, actions are important sources to change the dynamics in the system. An action changes agent's belief and causes other agents' beliefs to update through business connections. It is necessary for an eBBN model to direct represent and dertermine inference rules on agent actions. In this section, we discuss our methodology to introduce actions into eBBN model. This model extends $eBBN_1$ and is called $eBBN_2$. Firstly, we give a literate review on related works; secondly we define action nodes, links and related probability calculation rules for eBBN model;

thirdly we study the syntax and semantic of $eBBN_2$; lastly, we discuss the relationship among three eBBN models we define.

4.7.1 Previous works

In principle, actions are not part of standard probability theory. They represent perturb that disturbs casual relationship captured by the theory. In order to study the intervention of actions to a planning system such as STRIP and BURIDAN, a school of researchers [53] describe a "delta rule." The rule says each actions is the combination of a fixed set of atomic actions, denoted as do(p). do(p) is an independent elementary impulse that take effect on an observation variable p so that it will be changed from $\neg p$ to p in case the current state satisfies $\neg p$. In addition, one atomic action only takes effect in one observation variable but leaves everything else unchanged. The variants of this approach are embedded in STRIPS as well as other probabilistic planning systems. The problem of this rule is that they do not take into account the indirect ramification of actions. To handle such a problem, a causal theory of the domain, that specifies which event chains are likely to be triggered by a given action and how these chains interact when several actions concurrently occur in the system, needs to be constructed. The work of Dean and Wellman [54] shows the problem can be solved effectively using the language of causal graphs.

The semantics behind causal graphs and their relations to actions have been discussed in [55][56]. In [49], it gives several sound rules to inference the interventions of actions when the causal graph is not fully parameterized. In [51] a symbolic causal network, called action network that formalizes reasoning over actions under uncertain conditions is discussed. The network consists of controllable nodes and persistent nodes, which describe actions and its effect respectively. The direct effect of controllable nodes on persistent nodes follows “delta rule.” The micro-theory encodes the domain constraints and is used to solve the ramification and concurrency problem. The work [51] is a theoretical tool to support a planning system simulation.

The previous research works can be summarized as follows: actions are treated as direct cause of an observation variable with no pre-conditions. They are outside stimuli that cause beliefs in the system to update by adding new assertions. The indirect effect of actions propagates in the way that is similar to the propagation of a proved fact and passive observation or following special domain constraints. (See the below figure).

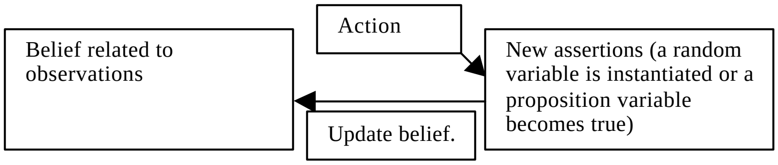

Action
Belief related to observations
New assertions (a random variable is instantiated or a proposition variable becomes true)
Update belief.

Figure 4.3 Actions and belief update

Since all of previous work listed above tries to embed actions into the planning systems, when these concepts are applied to the eBBN model for MASCM, they have several limitations. First, the domains are different. MASCM is a system that consists of multiple agents that do business on behalf of their owners. An action is triggered by agents' decision with internal business logic and usually occurs in certain order. Moreover, within a very short time period or at a certain time point, many agents may take actions. These actions might contain conflict goals. Thus, the eBBN model needs to specify rules on how to observe action consequences and solve the concurrent problem occurring in an open market system.

Second, actions that can change causal structure are not clearly studied. However, this type of actions is important to MASCM uncertainty analysis. The reason is the instance of sell-buy connection does not last forever. When one supplier fails to fulfill its commitment, the customer agent will find an alternative one. That is, the actual causal relationship between agents' belief on commitment failure changes over time.

4.7.2 Introduce actions into $eBBN_1$

In this section we introduce actions into $eBBN_1$. We first discuss the basic issues on how to define and observe agents' actions. Then, we introduce action nodes and action links into eBBN model. Lastly, we define conditional probability distributions for action nodes and its children nodes. Since an action is caused by agent's decision, in the rest of this dissertation, these two terms are interchangeable.

4.7.2.1 Action concepts

Based on the delta rule, one key idea of introducing action into causal graphs is to organize causal network into a few basic mechanisms, each involving a relatively small number of variables, and each disturbed by a certain group of actions. One group of actions only overrules one mechanism while leaving others unaltered. The impact of this group of actions on the whole system can be computed from the constraints between the directly affected mechanism and remaining ones [49]. In a $MASCM_1$, an agent is the natural and basic mechanism to study the disturbance from a group of actions. The impact of these actions propagates through links in the model $eBBN_1$. We clarify issues of action identification, description, ramification and concurrency based on this setting.

Action identification

According to their impacts to the causal relationship related agents' beliefs, we identify two types of actions. One type of actions is to set or change the relationship between agents belief on commitment failure; the other type of actions is to set agents' belief as accepted facts. In a $MASCM_1$, the example of the former type of actions is that a customer agent takes an action to switch the business partner during the negotiation process. In that situation, the current instance of sell-buy connection between two agents has been removed and a new one is established. The example of the latter case can be found when agents decide to finish the commitment, e.g. a customer agent cancels the previous order.

Action descriptions

Since "action" is a transitive concept, it only can be described through the consequence that made to a variable in the basic mechanism. This observable consequence is usually called an action's *intervention* (to a causal system).

To a causal network, an action's atomic and direct intervention is to add a new fact to the basic mechanism. It is equivalent to one or more propositions presented in the model being proved. That is, one or more nodes in $eBBN_1$ that associate with certain agents have been instantiated. Therefore, interventions of an action can be denoted as follows, "set($u_i = b$))," which says a random variable u_i is instantiated as b; or it could be *"idle,"* which represent zero intervention or no-intervention to the model. We say an action is in an "idle" (or "set($u_i = b$)") state if the intervention "idle" (or "set($u_i = b$)") of this action is observed.

Action ramification

Action consequence ramification describes how to deduce the indirect effects of actions. It is similar to solving the frame problem by deriving frame axioms from the completeness assumption of effect axioms [51]. To solve the ramification problem using domain constraints has been proved no more difficult than infer the effect from causal structure directly [51]. Therefore, similar to the work in [51], while modeling $MASCM_1$ as an eBBN, we use domain constraints to infer action indirect effects. In the MASCM, these domain constraints are more easily to implement as a social conventions that all agents agree to conform to. The constraints imposed on the action intervention ramification in eBBN model are as follows.

First, any actions that imposed on variables that associate with a customer agent will cause at least one group of variables that associated with one of its direct suppliers to reset.

- When a customer agent takes an action to give up the commitment to its own customer, all variables associated with its upper stream agents have to be reset.
- When a customer agent switch to a new product provider, variables that associates with the previous one have to be reset.
- When a customer agent cancels the order to one of its direct supplier, variables that associate with this supplier have to be reset.
- The reset will propagate from the lowest tier agent up to all agents currently in VSC till the raw material provider.

Second, the propagation of action effects to variables associated with an agent's downstream is in the same way as a proved proposition (related to this agent) does. This constraint is the direct application of Judea Pearl's inference rule over actions [49].

Intervention measurement and concurrency

Another issue related to actions' intervention is how we measure the impact of actions. In other words, at what time point, the action can take back from "set" state to "idle" state. In addition, how do we measure the consecutive actions occurring within a small time interval? In [52], new variables (nodes) are added to the network to represent a node's different values at different time point. This approach makes the causal network representation not consistent over time. To solve this problem, instead of updating model structure time by time, we post constraints on action observation. We assume, first, an action's intervention will "immediately" take effect. In other words, there is no measurable time points between the one when an action occurs and the one the node has been instantiated. Second, an agent's action returns to "idle" states when all nodes in the same mechanism (associated with the same agent) has updated their posterior probability according to the new fact discovery. If it happens in the same or different mechanism after the previous one is back to "idle" state, the action can be independently measured. Otherwise, its intervention is recognized but the consequence is though to be caused by the previous action. This action is thought to return back "idle" state as the same time point as the previous one does. Third, if there are concurrent actions in the system, their interventions are recognizable when they follow the domain constraints. That is, they can be combined together and treat as a single action. Otherwise, system is in an invalid states.

In some scenarios, several actions occur in a synchronized and ordered way. They consist of an action sequence or an agent's plan. For example, after an agent takes action to cancel the commitment to its customer, it will go ahead to stop any business activities with its direct suppliers inspired by that commitment. Usually, there is business logic behind the scene. To study the intervention of an action sequence from one agent, the action sequence is treat as a single action. That is, all of them will occur at the same time. And they will instantiate several nodes in the system simultaneously.

After we define how to measure an action (actions) intervention to the system, it can be seen that actions are thought to happen only when all others in "idle" state. When there is action a occurring that *really* changes system uncertainty, agent A_i's belief update on commitment failure before another action occurs is written as $p_a(x_i = 1 | E)$. (Probability $p(x_i = 1 | E)$ represents agent A_i's belief without any interference of actions or the belief in $eBBN_1$).

4.7.2.2 Action nodes and links

Based on the discussion in last section, we define actions nodes, links to describe agent's actions and their causal relationship with agent's belief.

Definition 4.17. *Action node* z_i is a special type of random variables defined in the causal network for the MASCM. The node maps one of agent A_i's actions a. This node does not have any parents and only impact one non-action node u_i associated with A_i. It takes action a's intervention "idle" and "set($u_i = b$)" as its value and only directly causes node u_i to change value. $< z_i, u_i >$ is called *action link*.

Action nodes are used to describe agents' autonomous actions. To some degree, they are higher-level abstraction of agents themselves. Thus, they can have arbitrary prior distribution and do not have any parent nodes. If an action takes non-idle interventions, its direct child node will be instantiated, which will further blocks any propagation between nodes in the direct child's parent set and its children set. Following rules in [49], for any descendents of a direct affected non-action node in the same basic mechanism, the effect of an action is equal to the passive observation of its instantiation; for the ascendancy nodes associated with the same agent, we define that the effect of an action is to reset them back to the initial statuses.

According to the intervention to causal relationship as we discuss in last section, action nodes are further categorized into two types. They are defined as follows,

Definition 4.18. Supplier selection decision variable v_{ji} only entails and direct disturb node l_{ji}. It takes one of action interventions set($l_{ji} = 1$), set($l_{ji} = 0$) and *idle* as values. If v_{ji}= set($l_{ji} = 1$), there are chain activities between agent A_j and agent A_i. If v_{ji}= set($l_{ji} = 0$), there is no sell-buy connection is to be setup between agent A_j and agent A_i.

The following figure shows the how to add node v_{ji} and related action link into $eBBN_1$,

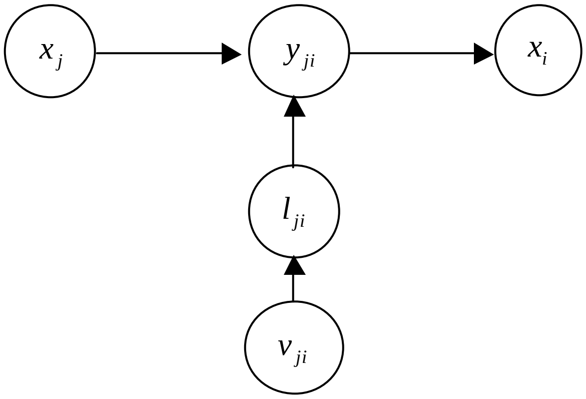


Figure 4.4 how to add node v_{ji} and its related action link into BBN

The semantic difference between node v_{ji} and node l_{ji} is that the formal represents the actual action source that causes certain observation occurred while the latter represents possible action consequence that an agent can observe.

Definition 4.19. Commitment fulfillment decision variable v_i only entails and directly disturbs node x_i. It can take one of action post-interventions set($x_i = 1$), set($x_i = 0$) and *idle* as its value. If $v_i =$ set($x_i = 1$), agent A_i decides to give up the commitment it currently holds. If $v_i =$ set($x_i = 0$), agent A_i has accomplished the commitment or guarantee its completeness to its customer agent.

The following figure shows the how to add node v_i and related action link into eBBN graph,

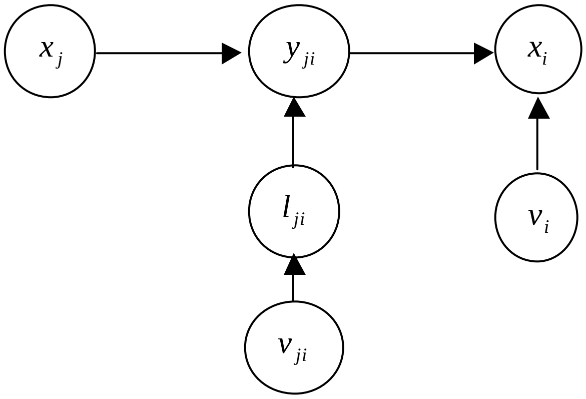


Figure 4.5 how to add v_i node and its related action link to BBN

When we consider the decision procedures related to chain activities or OFP between a pair of directly linked supplier and customer agents, we could find that an action to finish a commitment successfully or unsuccessfully is automatically followed by an action to change the status of existing demand-supply relationship but not vice-verse. For example, after an agent gives up current commitment to its customer, the next action is to cancel all orders to its customer. However, it can switch to better supplier without really finishing its commitment to the customer. According to this observation, any actions to finish a commitment will create an action sequence, which consists of action to finish a commitment and actions to remove all buy-sell connections with direct suppliers. The corresponding operation on actions nodes is whenever node v_i takes non-idle values, node v_{ji} ($A_j \in A_i^s$) take set($l_{ji} = 0$) as its value.

4.7.2.3 Probability distributions

In this section, we specify conditional probability distribution of nodes after action is introduced into the causal network. For nodes that do not have action nodes in their direct parent set, they have the same conditional probability distribution as in $eBBN_1$. We use symbol a to represent actions that disturbs agents' belief. This convention is through the rest of dissertation.

We have following definition.

The conditional probability table of node l_{ji} after action nodes and links are added into the model is as follows,

Definition 4.20.

$p_a(l_{ji}=1 \mid v_{ji}) = p(l_{ji}=1)$, if v_{ji} = “idle”;

$p_a(l_{ji}=1 \mid v_{ji})$ =1, if $v_{ji} = set(l_{ji}=1)$;

$p_a(l_{ji}=1 \mid v_{ji})$ =0, otherwise;

The conditional probability table of node x_i after action nodes and links are added into the model is as follows,

Definition 4.21.

$p_a(x_i=1 \mid p_i, v_i) = p(x_i=1 \mid p_i)$, if v_i = “idle”;

$p_a(x_i=1 \mid p_i, v_i)$ =1, if $v_i = set(x_i=1)$;

$p_a(x_i=1 \mid p_i, v_i)$ =0, otherwise.

After action nodes, action links and the related conditional probability distribution computation rules have added $eBBN_1$, we have extended the representation and inference capability of the network. The extended $eBBN_1$ model is called $eBBN_2$. Its formal definition is as follows,

Definition 4.22. $eBBN_2 = (V_2, EL_2)$, $V_2 = V_1 \cup \{v_{ji}, v_i \mid v_{ji}$ and v_i associates $A_i \in \mathrm{S}$, $A_j \in A_i^s\}$, $EL_2 = EL_1 \cup \{<v_{ji}, l_{ji}>, <v_i, x_i> \mid A_i \in \mathrm{S}\}$.

In the next section, we study $eBBN_2$ semantic and syntax.

4.7.3 Syntax and semantic analysis of model $eBBN_2$

From the discussion in last section, we can see, when we remove action nodes, action links and related attributes, $eBBN_2$ is identical to $eBBN_1$. Since each action node does not have parents and only directly entails one node, action links does not create any circle to the graph. Thus, for the syntax of $eBBN_2$, we have following theorem,

Theorem 4.5 $eBBN_2$ is a reverse tree.

Proof. The proof is directly from Theorem 4.4 and Definition 4.23.

□

After we define the model $eBBN_2$, at any given time, agent belief and uncertainty in system $MASCM_1$ can be determined using following theorem.

Theorem 4.6. $p_a(x_i = 1 \mid E) == 1 - \prod_{l_{ji}=1} (1 - e_{ji} p(x_j = 1 \mid E))$, $A_i \in \mathrm{VSC(S)}$.

Proof. At certain time point, model of $eBBN_2$ is in one of the following situations,

Case1,

At the time point, all of the action nodes in $eBBN_2$ take “idle” as its value. No actions affect the system belief. Obviously, model $eBBN_2$ is reduced to model $eBBN_1$. Thus, we have,

$$p_a(x_i = 1 | E) = p(x_i = 1 | E) = 1 - \prod_{l_{ji}=1}(1 - e_{ji} p(x_j = 1 | E)) \tag{4.15}$$

Thus, at this moment, the theorem holds.

Case2,

At one moment, node v_{ji} takes non-idle intervention as its value. According to the discussion in Section 4.7.2.1, it only directly affects links between commitment failure variable x_j and x_i. Any other nodes $x_k \in eBBN_2$ $(k \neq i)$ are indirectly affected following the ramification rules listed in Section 4.7.2.1 and their probability distributions can be determined by using the same equation in $eBBN_1$. That is, for any node $x_k \in eBBN_2$ $(k \neq i)$, $A_k \in$ VSC(S), we have,

$$p_a(x_k = 1 | E) = 1 - \prod_{l_{jk}=1}(1 - e_{jk} p(x_k = 1 | E)) \tag{4.16}$$

Considering node x_i and the ones with its direct supplier set, we do the similar reduction as we did in Theorem 4.2. Let pseudo-node X_q^k contains the node $L_i^k \cup V_{ji}^k \cup Y_i^k \cup p_i^k$, similar to the proving process before, we can have,

$$p_a(x_i = 1 | E) = 1 - \prod_{k=1}^{l}(1 - E_{qi}^{l} p_a(X_j^k = 1 | E)) \tag{4.17}$$

The E_{qi}^k represents the strength of pseudo link from X_q^k to x_i. Thus the value of it is the same as we defined in the process when we prove Theorem 4.4. When this action causes the node l_{ji} to be instantiated as 1, based on Definition 4.20, it can be seen that $p_a(X_q^k = 1 | E) = p(X_q^k = 1 | E) = p(x_j = 1 | E)$. Then, from Equation (4.17), we have,

$$p_a(x_i = 1 | E) = 1 - \prod_{l_{ji}=1}(1 - e_{ji} p(x_j = 1 | E)) \tag{4.18}$$

When node l_{ji} is instantiated as 0, the belief of agent A_j has no impact of A_i's belief. Equation (4.18) still holds. From Equation (4.16) and (4.18) shows, in case 2, theorem is proved.

Case3,

Assume at one moment, node v_i takes a non-idle intervention as its value. Similar to Case 2, we have,

$\forall x_k \in eBBN_2$ $(k \neq i$, $A_k \in$ VSC(S))

$$p_a(x_k = 1 | E) = 1 - \prod_{l_{jk}=1}(1 - e_{jk} p(x_j = 1 | E)) \tag{4.19}$$

For the relationship between node x_i and ones associated with A_i's direct supplier set, we have, (at this moment we also have $v_{ji} = 0$, for $A_j \in A_i^s$)

$$p_a(x_i = 1 | E) = 1, \text{ if } v_i = set(x_i = 1); \tag{4.20}$$

$$p_a(x_i = 1 | E) = 0, \text{ if } v_i = set(x_i = 0). \tag{4.21}$$

Equation (4.20) and (4.21) are a special case of Equation (4.19). Based on (4.19), (4.20) and (4.21), we say in case 3 the theorem is proved.

The proof of Case 1, 2 and 3 shows Theorem 4.6 holds.

□

Based on the above discussion, in $eBBN_2$, agents' belief on commitment failure can be determined by using the similar equation to the one in Noisy-Or networks.

4.7.4 Further discussion on model $eBBN_2$

From the discussion in last section we can see that intervention from an action only affects the syntax of the agent chain or DSDG. The supplier and customer relationship keeps intact. In other words, model of $eBBN_2$ is a direct mapping of a set of DSDGs for a $MASCM_1$. However, it is an implicit mapping of SDG. Compared $eBBN_1$ with $eBBN_2$, it is obvious that the former case is a time slice of latter one. That is, in principle, model of $eBBN_2$ has much more representation power than $eBBN_1$. Model $eBBN_0$ describes a special $MASCM_1$ ($MASCM_0$), which is a mapping of a set of DSDG that always equals to SDG. It is a special case of model. Put the discussion in Section 4.3.4, Section 4.4.3 together, we use following figure described the relationship of these three models according to their representation and inference power.

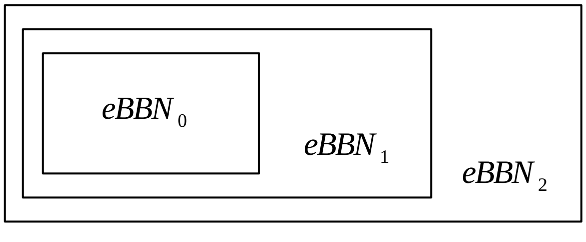


Figure 4.6 Relationships among models for $MASCM_1$

4.8 Summary

In this chapter, we formalize agents' interaction in an uncertain environment as a causal network. We define three eBBN model, $eBBN_0$, $eBBN_1$ and $eBBN_2$ for a $MASCM_1$, which have increasing representation and inference power. We also prove three model have similar syntax and semantics.

Chapter 5 Problem-solving Using eBBN Model

When an eBBN model is established for a MASCM, the model can be used as the basis to design algorithms that help agents deal with uncertainty. We identify the following three problems as examples for our discussion.

- Evaluate the impact of outside uncertain factors and take a corresponding action to commitment processing;
- Select or switch to a promising direct supplier.
- Search for the most critical or fragile suppliers in a Virtual Supply Chain (VSC).

In the first two problems, information stored or retrieved fom the eBBN model is used in functional agent decision-making procedures. In the last case, eBBN is used for informational agents to identify the weakness link in a formed VSC.

To simplify the study, we assume the underlying framework is a $MASCM_1$, and its $eBBN_2$ model is constructed. This chapter is organized as follows: firstly, we briefly review the Order Fulfillment Process (OFP) and the corresponding $eBBN_2$ operations; then, for each problem we present our algorithms.

5.1 OFP and its corresponding $eBBN_2$ Operations

Business activities between two (functional) agents are Order Fulfillment Process (OFP) as mentioned in Chapter 2. In this section, we briefly re-state this process, and then discuss corresponding $eBBN_2$ operations.

5.1.1 A review of Order Fulfillment Processes (OFP)

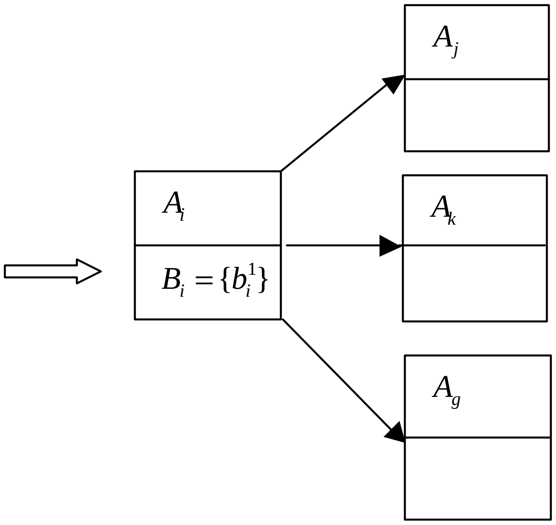
General speaking, an OFP can be logically divided into a sequence of steps. First, based on the commitment made to its own customers, a customer agent generates an order to each selected suppliers. An OFP *starts*. Since the fulfillment of a commitment usually relies on more than one material, in each OFP, the customer agent may generate more than one order.

Then, after negotiating with a desired supplier, one order is *temporally solved* if there is a commitment being reached between the customer agent and one of its suppliers. If the product is delivered to the customer agent, the order is *eventually solved*. If an order does not have a temporal solution (no commitment is made between the agent and its desired direct supplier) or the order is not eventually solved (commitment is aborted by its supplier or cancelled by customer agent), the customer agent has to find the alternative supplier to finish the task or has given up the commitment it holds. If it is the customer agent that actively de-commits previous agreement, the supplier agent has to cancel all orders that have been generated but are not eventually solved so far. A pair of customer and supplier agents notify each other the newly development related to the commitment all the times

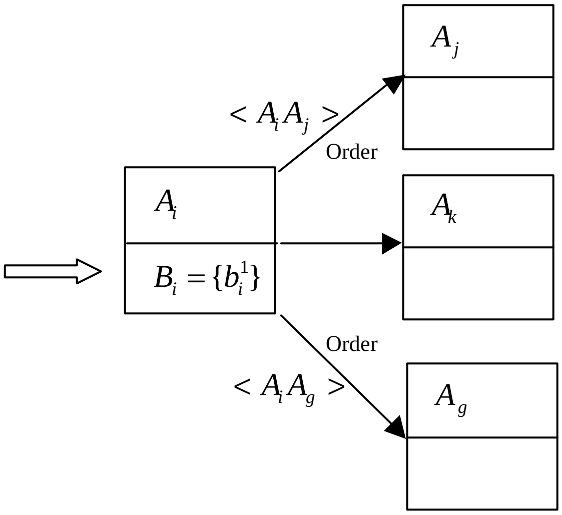
Finally, when direct suppliers provide all the material the customer agent needs, the commitment held by the customer agent is *resolved successfully*. Otherwise, the commitment is *resolved unsuccessfully*. In either case, the commitment held by this customer agent has reached a stable status. We say the OFP initialized by the customer agent is *completed*.

A virtual chain in the MASCM constitute of functional agents involved in multiple inter-connected OFPs triggered by the end customer internal needs, which represent as *the end order* to the system. The complete VSP is a solution to this particular end order.

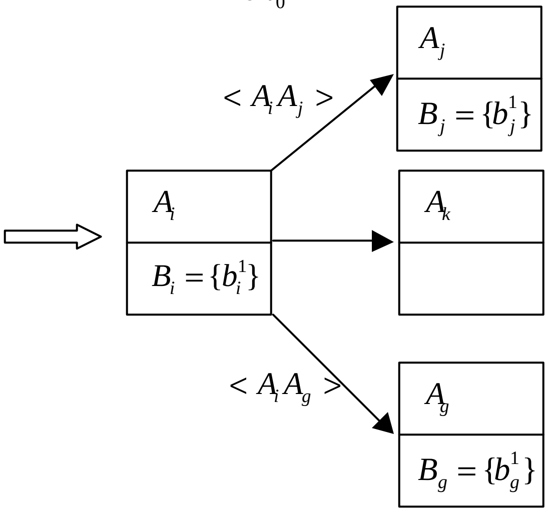
The following figure shows the OFP between an agent A_i and its direct suppliers A_j, A_k and A_g (G (A_k)=G (A_j)) at different time points till the commitment B_i held in agent A_i has been resolved completely. (All symbols used here follow conventions mentioned in Chapter 4; A_i, A_j, A_k and $A_g \in MASCM_1$).



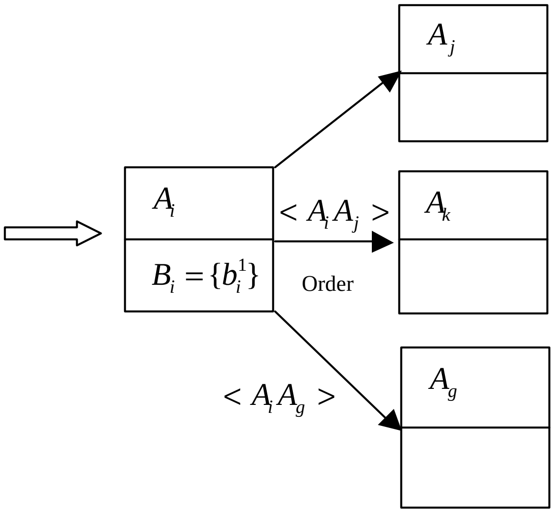
Time t_0



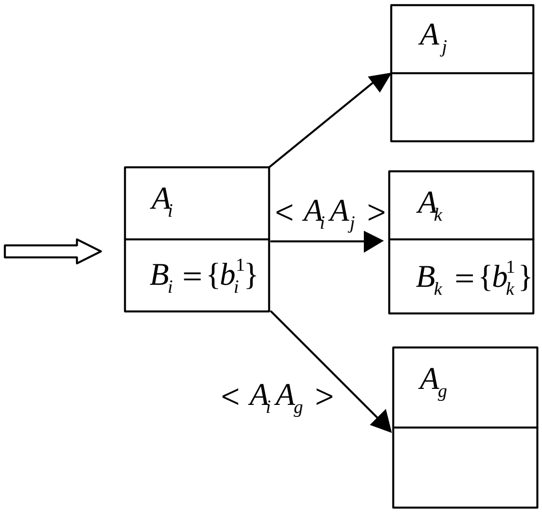
Time t_1



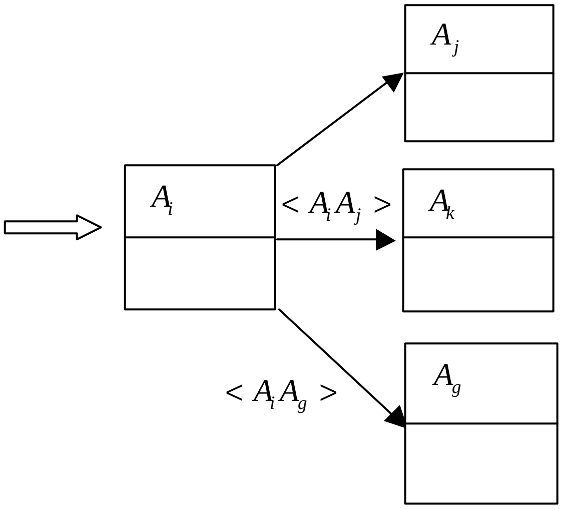
Time t_2



Time t_3



Time t_4



Time t_5

At time point t_0, agent A_i generates a commitment b_i^1 to its customer. An OFP starts between this agent and its direct suppliers. In order to accomplish this commitment, it generates orders to its supplier agents A_j and A_g at t_1. Both of suppliers make commitments with A_i at t_2. However, the commitment b_j^1 held in A_j fail at time t_3. Agent A_i sent another order to its supplier A_k that can provide the same product as agent A_j. At t_4, agent A_k made a commitment with agent A_i at the same time, agent A_g complete the commitment. At t_5, agent A_k finishes its commitment, so does agent A_i. The OFP is solvable. Agent A_i, A_k and A_g are part of an evolving VSC.

Figure 5.1 An example of OFP triggered by agent A_i.

5.1.2 Operate mode of $eBBN_2$

In the following sections we discuss how the model $eBBN_2$ evolves as the underneath system $MASCM_1$ is running.

5.1.2.1 Operate $eBBN_2$ units

Each functional agent operates the basic mechanism of $eBBN_2$ defined in Chapter 4. That is, each functional agent A_i in $MASCM_1$ independently manages local network consisting of nodes x_i, y_{ji}, l_{ji}, v_i, v_{ji}, links among them and their (conditional) probability distribution, e.g., link strength e_{ji} ($j \geq 1$) and agent initial belief on commitment failure, $BEL(x_i = 1)$. In this section, we discuss how a functional agent A_i operates $eBBN_2$ unit when it is involved in an OFP.

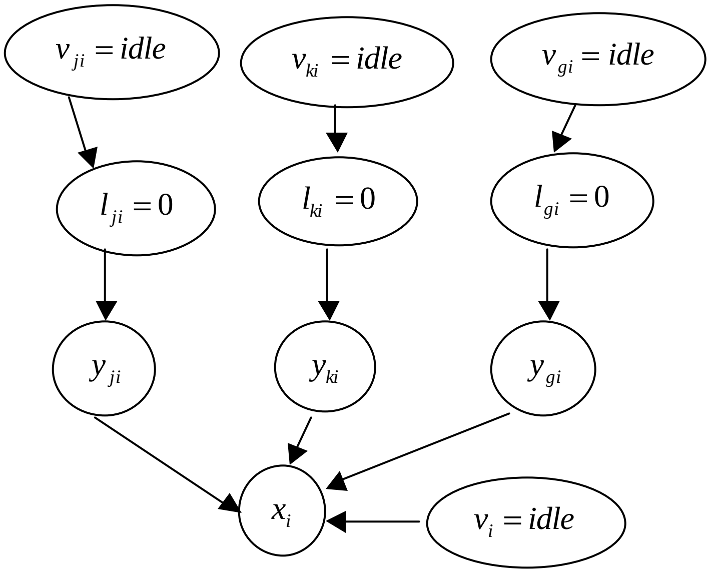
In the network, initially (or at any time system is reset), l_{ji}, v_{ji} are instantiated as 0 and "idle" respectively. All other nodes are un-instantiated. If possible, functional agent A_i updates its local knowledge about the outside, e.g. the value of e_{ji} and $BEL(x_i = 1)$ or $p(x_i = 1 | E)$ (E is an empty set during updating period).

When functional agent A_i generates an order to its direct supplier agent A_j, an OFP starts. Correspondingly, the node v_{ji} is instantiated as "set (l_{ji}=1)" and instantly, l_{ji} is instantiated as 1. Within a certain small time interval, after other nodes associated with A_i update their posterior probability, v_{ji} is back to "idle" state. The direct effect of node l_{ji}'s instantiation is that the probability distribution of dummy node y_{ji} is equal to node x_j's distribution till the negotiation fails or the commitment has been resolved. During this period, both agent A_i and agent A_j may exchange information and update their beliefs. The message from agent A_j to A_i describes agent A_j performance estimation on commitment made with A_i. The message from agent A_i to A_j indicates agent A_i's desire for agent's performance. Rules for agent A_j (A_i) to update their units are in next section.

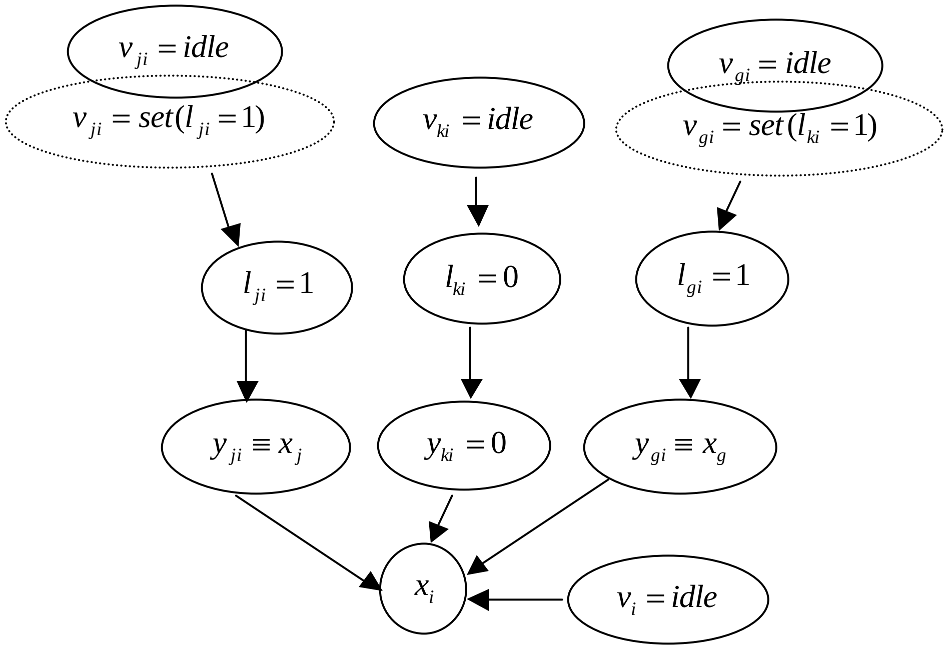
When negotiation fails or functional agent A_j gives up the commitment, agent A_i instantiates node v_{ji} as "set (l_{ji}=0)" so that both node l_{ji} and node y_{ji} is re-instantiated as 0 again. At the same time, mechanisms that associated with agent A_j and its upper stream agents (A_j^u) will reset. However, if agent A_i decides to give up its commitment to the customer, it instantiates node v_i as "set(x_i=1)" and node v_{ji} as "set (l_{ji}=0)." (This is an action sequence). Correspondingly, nodes x_i and l_{ji} are instantiated as 1 and 0, respectively. These instantiations will cause nodes associated with A_i^u to reset and they will propagate to nodes associated with A_i^d .

When all of agent A_i's suppliers successfully resolve the commitments to A_i (if there indeed has one between them), the OFP initialized by agent A_i completes. Agent A_i instantiates node x_i as 0 and propagate it to nodes associated agents in A_i^d . When an end order has been eventually solved, the complete VSP consists of agent A_i that associates with the node x_i=0.

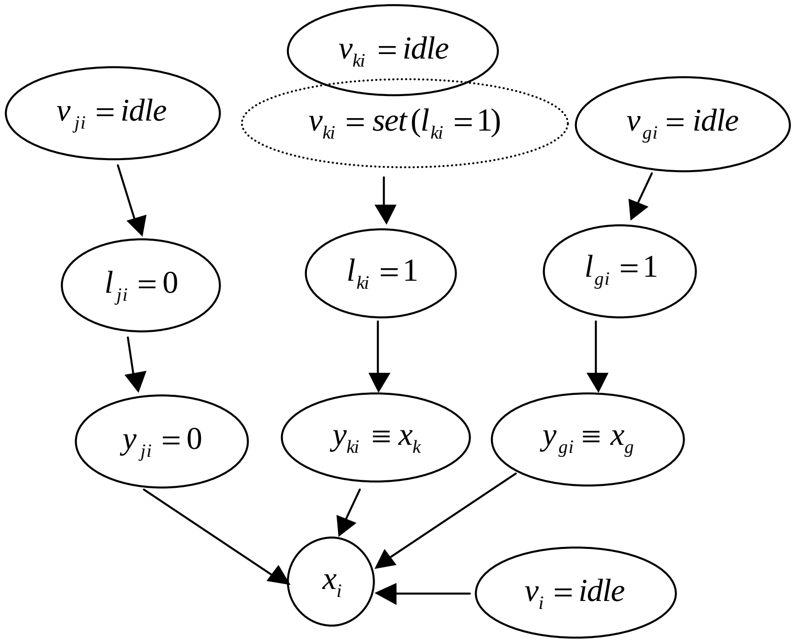
The following figure shows the A_i's operation on (part of) $eBBN_2$ according to the example in Section 5.1.1.



Time t_0



Time t_1 and t_2



Time t_3 and t_4

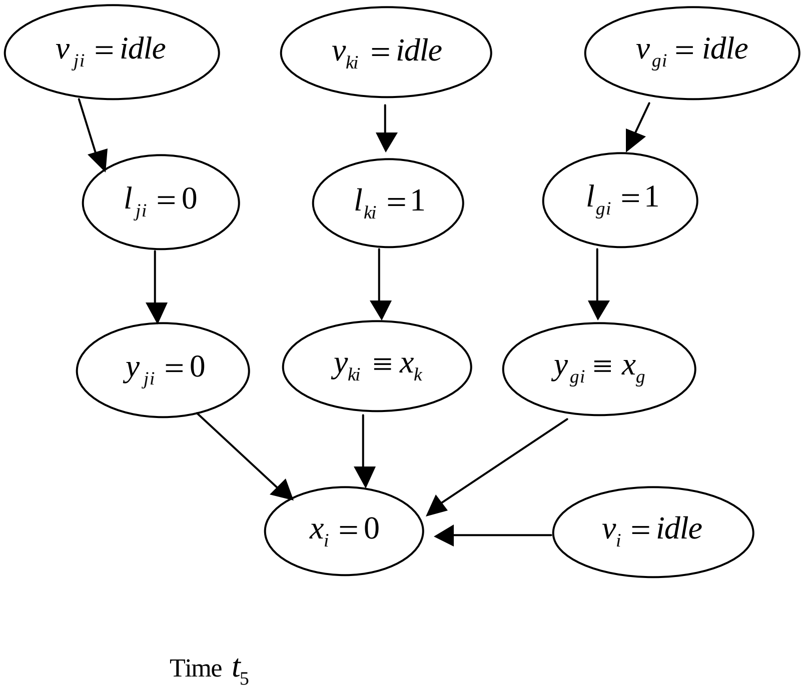


Figure 5.2 An example of local operation of $eBBN_2$ in agent A_i.

5.1.2.2 Update an $eBBN_2$ unit with probability propagation

In last section, we can see while there is an agent interaction in OFP, there exists functional agent belief propagation and probability update in corresponding $eBBN_2$ units. In the last chapter, we have proved that model of $eBBN_2$ has a reverse tree structure (Theorem 4.5) and can be seen as a Noisy-Or network that consists of units independently operated by individual agents (Theorem 4.6). Therefore, rules on agents' belief update and propagation are similar to ones in a singly connected Noisy-Or network (see Chapter 2). In following paragraphs, we only explain how to apply belief update and propagation rules in $eBBN_2$. In this context $p(x_i = 1 | E)$ ($p(x_i = 0 | E)$) has the same meaning as $BEL(x_i = 1)$ ($BEL(x_i = 0)$), which represents a functional agent A_i's belief on a commitment failure (success).

First, if functional agent A_j is one of functional agent A_i's suppliers, we use the symbol Π_{ji} and l_{ij} to represent the propagation message from the unit that contains node x_j to the one contains x_i through y_{ji} (from a parent x type node to a child x type node) and reverse direction respectively. The content of propagation message agents generate follows the standard the type of Π (l) message format. That is, the type of Π message from the unit operated by agent A_i to the one operated by agent A_j is the current probability distribution of node x_j. It is denoted as follows,

$$\Pi_{ji} = [\, p(x_j = 1 | E)\, , p(x_j = 0 | E)\,]. \tag{5.1}$$

Since $p(x_i = 1 | E) + p(x_i = 0 | E) = 1$, only one of these values is necessary to propagate. We usually set $\Pi_{ji} = p(x_i = 1 | E) = BEL(x_i = 1)$ as the message content that propagates from a parent node to its child node.

The type of l message in the reversed direction is denoted as $l_{ij} = [\, l_1, l_0 \,]$. Followed the discussion in [41][40], it can be determined using following equations,

If node x_i is not instantiated,

$$l_1 = 1,\ l_0 = 1. \tag{5.2}$$

If node x_i is instantiated as 0,

$$l_1 = (1 - e_{ji})\prod_{k \neq j}(1 - e_{ki})\Pi_{ji},\ l_0 = \prod_{k \neq j}(1 - e_{ki})\Pi_{ji}. \tag{5.3}$$

If node x_i is instantiated as 1,

$$l_1 = 1 - (1 - e_{ji})\prod_{k \neq j}(1 - e_{ki})\Pi_{ji},\ l_0 = 1 - \prod_{k \neq j}(1 - e_{ki})\Pi_{ji}. \tag{5.4}$$

Second, when agent A_i receives a p_{ji} message, it updates probability distribution of node x_i through the following equation,

$$p(x_i = 1 \mid E) = s\, l_1 \cdot (1 - \prod_{l_{ji} = 1}(1 - e_{ji}\Pi_{ji})). \tag{5.5}$$

$$p(x_i = 0 \mid E) = s\, l_0 \cdot (1 - \prod_{l_{ji} = 1}(1 - (1 - e_{ji})\Pi_{ji})). \tag{5.6}$$

Third, after agent A_i updates its belief, if no actions, probability distribution of other nodes in the same unit is not changed and agent A_i propagates the belief to the unit operated by its directed customer agent. At the same time, it generates and sends l_{ji} to its direct supplier agent A_j. If actions happen, certain nodes are instantiated. These instantiations propagate to units controlled by downstream agents; however, they will cause all units operated by its direct suppliers to reset as we discussed in Section 5.1.2.1.

Thus, no λ messages would be generated and broadcast in $eBBN_2$. The agent belief updated described in equation (5.5) and (5.6) can be further rewritten as follows,

$$p(x_i = 1 \mid E) = (1 - \prod_{l_{ji}=1} (1 - e_{ji}\Pi_{ji})) ; \tag{5.7}$$

$$p(x_i = 0 \mid E) = (1 - \prod_{l_{ji}=1} (1 - (1 - e_{ji})\Pi_{ji})) \tag{5.8}$$

In the model, we require a message, l' (in this case l'_{ji}), which follows Equation (5.3), is generated and propagated one tier up to the units that are associated with supply agent A_i^s, whenever distribution of node x_i changes and provided it is not instantiated by agent actions. This information is used model the process that direct supplier agent A_j evaluate agent A_i's requirements. This information helps supplier agent understand customer needs so that a high quality service can be provided. In addition, if a supplier agent feels the customer request more than it can afford, it proactively terminates the service and put its service capability to other orders. The detailed discussion is in section 5.2. It also can be noticed that, when actions are not concerned in agent belief analysis, Equation (5.1) through (5.6) is applied. They are used for informational agents to generate public service for functional agents. The details are in Section 5.4.

The following algorithm summarizes agent A_i's local operation on $eBBN_2$ when it deals with its direct supplier agent A_j,

Algorithm 5.1:

```
Agent-Operations-on-Local-Network ( $x_i$ , $y_{ji}$ , $l_{ji}$ , $v_{ji}$ , $v_i$ , $A_j$ )
if $v_i$ == "set( $v_i$ =0)"
    then instantiate $x_i$ as 0;
        reset nodes associated with $A_i^u$ ;
        propagate $BEL(x_i = 1)$ to $A_i^c$ ;
        return;
if $v_i$ == "set( $v_i$ =1))"
    then instantiate $x_i$ as 1;
        $v_{ji}$ = "set( $l_{ji}$ =0)";
        propagate $BEL(x_i = 1)$ to $A_i^c$ ;
if $v_{ji}$ == "set( $l_{ji}$ =0)"
    then instantiate $l_{ji}$ as 0;
        instantiate $y_{ji}$ as 1;
        reset nodes associated with $A_j^u$ ;
        return;
if $v_{ji}$ == "set( $l_{ji}$ =1)"
    then instantiate $l_{ji}$ as 1;
        $y_{ji} = x_j$ ;
if $l_{ji}$ ==1 and there is a $\Pi$ message
    then based on Equation (5.1), update $BEL(x_i = 1)$ ;
        propagate $BEL(x_i = 1)$ to $A_i^c$ ;
        if $\Pi$ message is not from $y_{ji}$
            then Based on Equation (5.4), generate and
  send $l'_{ji}$ type of message to $A_j$ .
```

Algorithm 5.1 shows how an agent generates, retrieves and infers uncertainty information from $eBBN_2$. This information can be used in an agent's decision-making processes, which is discussed in following sections.

5.2 Problem 1: decision on commitment cancellation with uncertainty

In this section, we study how agents utilize uncertain information from local $eBBN_2$ network unit to make decision on Commitment Cancellation. We first describe the scenario, and then present our algorithms.

5.2.1 Commitment Cancellation with uncertainty

In the business environment unexpected events happen at a high frequency. A company has to face this challenge and adjust strategies accordingly. Since the commitment is the vital connection in the transaction between companies, one crucial decision the manager has to make is to see whether or not to cancel the commitment when the situation changes. Consider the supply-demand relationship between Personal Computer (PC) manufacturer and CPU chip provider, e.g. the relationship between DELL and Intel. When Intel announces that a new type of mobile chip will be available in the market shortly, DELL decides to use it in its laptop product line. After several rounds of negotiations, DELL and Intel sign a contract. But in the fourteenth day before the delivery date, a piece of information from Intel says the new chip may delay ten days in the market. In other words, the contract (the written commitment) might have certain chance to fail. Now, the managers in DELL have to use this information to make its next move, to stick with contract or just cancel the order. At this moment, it has to evaluate the possible losses and gains from uncertain factors. When facing the challenge of uncertainty, the decision about commitment processing, which is similar to managers in DELL have to make, is called *Commitment Cancellation*. Since currently no existing methods can quantitatively measure information accompanied with uncertain factors, this type of decisions is based on the managers' experience and estimation. The result usually is inaccurate and unconvincing. In next two sections, we present two algorithms based on $eBBN_2$ model for agents' Commitement Cancellation. In the first algorithm, we design an Expected Utility Function of Commitment (EUFC). In the second approach, agents directly use the information sent from its customer (l'_{ji}) to make decisions.

5.2.2 Commitment Cancellation with expected utility

When detailed information that describe current environment is available, there are many approaches that can help managers in a company to make decisions. Within them, in recently years, Multi-Value Utility Function (MVUF) has won wide popularity [1]. This approach gives a reference to next-step actions over multiple concerned parameters in different practical business settings through the utility value calculation [1]. Studies shows now 80% of general managers in Fortune 500 companies are implicitly or explicitly use MVUF as their on-hand quantitative analysis tool to make decisions [57]. In this context, we assume the MVUF is the approach used for agent automatic decision-making. That is, each functional agent contains a set of internal utility functions. Agents use these functions and their calculated value as the only references to take actions. In this section we show when there are only uncertain information available, how to transform a MVUF to an Expected Multi-Value Utility Function (EMVUF) based on the agent beliefs propagated in model $eBBN_2$. To carry on our study, we assume each agent A_i has a Utility Function of Commitment (UFC) u_i. It can compute utility (gain and loss) according to the commitment that an agent current holds (to its customer). This utility is used for agents to make decision on Commitment Cancellation.

5.2.2.1 Utility Function of Commitment (UFC) u_i

How to design a MVUF is beyond the scope of this dissertation. In this section the UFC we use is over four factors, the selling price, the manufacturing and transportation cost, inventory management expenses and the benefits from the improvement of customer satisfaction. This function u_i, associated with agent A_i, could be expressed in the following equation,

$$u_i = P_i - C_i - I_i + S_i \tag{5.9}$$

In this equation, symbol P_i stands for the first factor, the negotiated price of the service agent A_i provides to its customer; symbol C_i stands for the second factor, the manufacturing and transportation costs when agent A_i produces the product; symbol I_i stands for the third factor, the inventory expense. It is the management charge for agent A_i as it reserves certain lever manufacturing power and materials for the quick response to the customer order; symbol S_i stands for the last factor, the estimation of future benefits when agent A_i successfully resolves the commitment made with its customer agent. In other words, S_i represents the potential benefits such as positive image establishment, market competition enhancement and so on when customer agent satisfies its service. The utility of this equation represents the profit that an agent can earn from fulfillment of a commitment (to its customer agent), which is measured by unit of certain currency, e.g. U.S dollar. In the equation, factors C_i and I_i represent an agent's expenses in the OFP. When these amounts increase, the profit will decrease. Thus, there are "–"

before these two symbols. On the contrary, the plus (+) symbol are put before S_i and P_i. When the agent knows loss and gain from the individual product it purchases from the suppliers, the equation can be further rewritten as follows,

$$u_i = P_i - C_i - \sum_{A_k \in A_i^s} I_k + \sum_{A_k \in A_i^s} S_i + S_0 \qquad (5.10)$$

In equation (5.10) symbol S_0 stands for the gain from the current customer satisfaction reach, for example, the benefit for setting up the long-term business relationship between these two agents.

Both Equation (5.9) and (5.10) are composed of the basic elements that may be concerned when an agent computes the utility related to the commitment made to its customer agent. Each agent might emphasize on different factors through adding a weight value for each of them. In addition, both functions are examples that can be used in agent Commitment Cancellation process. Human owners can design and implementation different ones for their agents.

5.2.2.2 Expected Utility Function of Commitment (EUFC) $E(u_i)$

In this section we discuss how to transform UFC to Expected Utility Function of Commitment (EUFC) that can utilize information propagated in $eBBN_2$ for Commitment Cancellation in an uncertain environment.

As we discuss in Section 5.1, both of the supply and demand sides locally operate the $eBBN_2$ units. They propagate the commitment process failure probability down to its direct connected nodes through sending Π type of messages. If agent A_i receives a Π type of message, it will update probability distribution of node x_i using Equation (5.7) and (5.8) respectively. The updated agent belief can be used as an estimation of the gain or loss of current deal made with the customer. For instance, the production of $p(x_i = 0 | E)$ ($BEL(x_i = 0)$) and the negotiated price P_i is a good metric to measure the possible gross income from this deal. Moreover, the commitment processing information from the parent nodes (supplier agent's belief) can be directly used to compute the utility change. For example, through agent belief propagation, agent A_i knows its direct supplier A_k has the possibility of $p(x_k = 0 | E)$ (this value can not be the real agent belief of agent A_k. It might be changed by agent A_k to protect its privacy) to finish the commitment. With the condition that if this material indeed arrives on time, the inventory management cost is I_k, the estimation of expense could be measured by the product of $p(x_k = 0 | E)$ and I_k. Similarly, the expected gain from the manufacturing power reserving to improve the quick response for customer order can be calculated by the product of $p(x_k = 0 | E)$ and S_k. The estimation of direct gain from the successful accomplishment of the commitment to the customer can be estimated by the production of $p(x_k = 0 | E)$ and S_0. As we combinate them all together, a UFC becomes an Expected Utility Function of Commitment (EUFC). The following equation summarizes above discussion.

$$E(u_i) = p(x_i = 0 \mid E)(P_i - C_i) - \sum_{x_k \in A_i^s}^{l} p(x_k = 0 \mid E) I_k + \sum_{x_k \in A_i^s}^{l} p(x_k = 0 \mid E) S_k + p(x_i = 0 \mid E) S_0 \tag{5.11}$$

5.2.2.3 Commitment Cancellation decision with EUFC

The computed result of EUFC is called the Expected Utility of Commitment (EUC). It can be used in an agent's Commitment Cancellation. To complete the procedure, an internal Commitment Threshold (CT) is needed, which describes an agent's the bottom line of minimum gain (utility) from the current commitment. When EUC is less than CT, the agent can go ahead to give up its own commitment to the customer and cancel the order to all its suppliers. This decision procedure could help agents to avoid further loss when there is a sign from the supplier side that indicates the commitment has high failure possibility. The procedure on Commitment Cancellation can be described as the following algorithm,

Algorithm 5.2:

```
Agent-Commitment-Cancellation-With-EUFC
($eBBN_2$ unit associated with $A_i$, $E(u_i)$, CT)
Update the distribution of $x_i$ and update $e_{ji}$ if necessary
Compute EU of $E(u_i)$
if $EU < CT$
        then $v_i$ =“set($v_i$ =1))”;
            Agent-Operations-on-Local-Network
            ($x_i$, $y_{ji}$, $l_{ji}$, $v_{ji}$, $v_i$, $A_i^c$);
            for $A_j$ in $A_i^s$
                $v_{ji}$ =“set($l_{ji}$ =0)”;
                Agent-Operations-on-Local-Network
                ($x_i$, $y_{ji}$, $l_{ji}$, $v_{ji}$, $v_i$, $A_j$);
            return
```

10. **else** go to 1.

Even though Algorithm 5.2 only describes agent decision procedure on Commitment Cancellation. However, by providing different EMVUFs and threshold values, algorithms for other types of decisions in an uncertain environment can be similarly constructed.

5.2.3 Commitment Cancellation with customer requirement evaluation

In last section, we show how an agent uses the information from upstream agents to make decision on Commitment Cancellation. In this section we show an agent also can utilize information retrieved from network associated with down stream agents to make similar decision with uncertainty.

In a traditional singly connected BBN, the message from a node to a parent node is generated only when there is new evidence observed by the descendent nodes. In model $eBBN_2$ new evidence discovery (a x type node is instantiated) is equal to commitment accomplishment (a commitment is resolved). In other words, the uncertainty in the partial system consisting of this agent and its upstream is no longer important for the end order fulfillment. It is the reason that a system level reset signal takes places of the l type messages.

However, that there are no l type messages in the model does not mean the customer agent does not initialize interaction with its suppliers. In fact, both of two types of information flows defined in Chapter 2 can initiate from a customer agent. In information flows, a customer notifies the direct supplier the attitude regarding to its current performance. This information is important for an agent decision-making in OFP under uncertainty. At the negotiation stage, it allows the supplier agent could evaluate customer agent requirements and move forward to sign the contract; at the commitment process monitoring stage, the supplier agent can use this information to improve its performance, e.g. organize more much facilities to finish the order considering the customer urgent situation. The supplier agent relies on this information to estimate customer requirement about the commitment.

In the $eBBN_2$, a $l^{'}$ type message is defined by Equation (5.3) and is generated and sent to its ascendant unit when the customer agent's belief is updated. It can be explained as the performance expectation from the customer agent to a direct supplier agent when customer agent's belief changes. It models information flows started by customer agents. A supplier agent can direct use this information to make decision on Commitment Cancellation when it finds the customer desire is beyond it can afford. In this way, the diagnosis inference capability of a causal model has been used. In next sections, we present an algorithm that uses $l^{'}$ type message for agents to cancel the commitment made with its customer.

5.2.3.1 Customer Desired Belief

Similar to the equation that is used to update parent node probability in a Noisy-Or network [41], when agent A_j receives a message l'_{ji}, instead to update the new distribution of node x_j, it update the *Customer Desired Belief* (CDB) using the following equation,

$$\text{CDB}= (a((1-e_{ji})\prod_{k\neq j}(1-e_{ki})\Pi_{ji})BEL(x_j=1)), a(\prod_{k\neq j}(1-e_{ki})\Pi_{ji})BEL(x_j=0))) \quad (5.12)$$

(a is the normalization parameter.)

We usually denote $a((1-e_{ji})\prod_{k\neq j}(1-e_{ki})\Pi_{ji})BEL(x_j=1))$ as CDB^1 and CDB^2 for $a(\prod_{k\neq j}(1-e_{ki})\Pi_{ji})BEL(x_j=0))$. Thus, CDB can be easily written as a non-order set as (CDB^1, CDB^2). The value CDB^1 and CDB^2 are the supplier performance improvement that the customer agent desires. By knowing either CDB^1 or CDB^2, the other one can be determined. Usually, we use CDB^1 to represent agent's requirements.

5.2.3.2 Commitment Cancellation with CDB

The value CDB^1 implies the quality of the service that the customer agent wants supplier agent to provide. The percentage of its change over a unit time period indirectly reflexes the customer need that a supplier agent can afford. The following algorithm shows how this percentage can be used in agent decision on Commitment Cancellation.

Algorithm 5.3:

```
Agent-Commitment-Cancellation-With-CDB
($l_{ki}$, $A_k \in A_i^c$, $CDB^1$, $A_i^s$, $\beta$)
update $CDB^1$ to $CDB^{1'}$ when receives $l_{ki}$ from $A_k$;
if $|CDB^{1'} - CDB^1|/CDB^{1'} < \beta$
    then $v_i$ ="set($x_i$ =1)";
    Agent-Operations-on-Local-Network
        ($x_i$, $y_{ki}$, $l_{ki}$, $v_{ki}$, $v_k$, $A_k$);
    for $A_j$ in $A_i^s$
        $v_{ji}$ ="set($l_{ji}$=0)";
        Agent-Operations-on-Local-Network
            ($x_i$, $y_{ji}$, $l_{ji}$, $v_{ji}$, $v_j$, $A_j$);
else $CDB^1 = CDB^{1'}$;
return.
```

In Algorithm 5.3, the constant β is to use as a threshold for the functional agent to decide whether is the time to stop the transaction with customer agent. Its value is in the range between 0 and 1 and is set before the OFP begins.

5.3 Problem 2, supplier selection and switch

In this section we discuss another problem-solving scenario: a functional agent selects and switches direct suppliers. In a $MASCM_1$, each agent may have more than one supplier agent that can provide the same product. Therefore, when an agent wants to generate an order to suppliers, it needs to choose the one that could best serve its interests. To select a reliable supplier lowers the chance of commitment failure. In addition, through the interaction with its suppliers, an agent might find its initial selection is not good enough. In this case, agents would like to compare supplier performance again and switch to a "promising" supplier. In this section we discuss two algorithms that use an agent's beliefs in eBBN model to make decision. One is used before the functional agent starts the negotiation; the other one is used either during the negotiation period or agents' chain performance monitoring stage.

5.3.1 Select a promising supplier before negotiation

In the business environment, a company needs to consider the supplier's reputation, evaluate the time constraints and refer to its past experience before the real negotiation starts. This review procedure protects a company from profit loss. Software agents follow the same rules. That is, before an agent generates an order, it has to spend some time to investigate the business record of a direct supplier agent. A functional agent A_i stores this type of knowledge about direct supplier A_j in link strength e_{ji}. $BEL(x_j = 1)$ received in previous transaction represents agent A_i's past experience with A_j. In addition, following the discussion in Chapter 4, we can see that the production of e_{ji} and $BEL(x_j = 1)$ can be explained as agent A_i's estimation of agent A_j performance. The smaller of this production is, the more likely that the agent A_j will keep its promise after it signs a contract. Therefore, before agent A_i starts negotiating process with an agent group for certain product, it can first do local computation for all the suppliers in this group to determine production value. Then it chooses the direct supplier with the minimum value as its negotiation partner (a potential supplier). The following algorithm gives a summarization of the discussion above,

Algorithm 5.4:

Agent-Select-Supplier-Before-Negotiation

```
$(g_k, \{e_{ji}\}, \{BEL(x_j=1)\}, A_i^s)$
min=$\infty$, k=-1;
for each $x_j$ that associates with agents in the set $\{A_j |$
    $g_k$=G($A_j$), $A_j \in A_i^s\}$
        if (min $> e_{ji} * BEL(x_j=1)$ )
            min=$e_{ji} * BEL(x_j=1)$
            k=j;
$v_{ki}$ ="set($l_{ki}$=1)";
Agent-Operations-on-Local-Network
    ($x_i$, $y_{ki}$, $l_{ki}$, $v_{ki}$, $v_k$, $A_k$);
for $A_j$ in $A_i^s$ and $j \neq k$
        Agent-Operations-on-Local-Network
        ($x_i$, $y_{ji}$, $l_{ji}$, $v_{ji}$, $v_j$, $A_j$);
return
```

5.3.2 Switch supplier dynamically

A more advanced $eBBN_2$ application in agent provider selection is that it can guild the agent to switch to a supplier with better performance dynamically after two agents have begun the negotiation but before the agreement has been resolved.

Let's first reconsider the example discussed in Section 5. When both Intel and AMD claim there will be a new type of chip at speed of 2G available in the market in one month, based on utility analysis, managers in DELL thinks embedding Intel chips into their new developing laptop will bring more profits than using AMD chips does. After several rounds of negotiation, DELL and Intel sign a contract. But in the fourteenth day before the delivery date, a piece of information from Intel says the new chip might be delay ten days to the market. In other words, the contract (the written commitment) might have certain chance to fail. Now, the managers in DELL have to use this information to make its next move. The decision can be either to stick with Intel (because there are still chances for Intel to deliver the chip on time) or just cancel pervious order to Intel then turns to AMD. If their choice is still with Intel, this new type of laptops may delay to market and the company might lose some of its old customers. The delay can cause inventory to increase as well. However, if they choose to switch to AMD, they have to re-negotiate with it and re-design the product line. This lead to extra operation costs for DELL. How to decide that which supplier is most suitable to deal with when there is a piece of uncertain information is critical for agents to make profits.

At the beginning of this chapter, we describe how an agent locally operates $eBBN_2$ units. It can be seen that if there is a negotiation process going on between two agents (supplier and customer), agents' beliefs have already been propagated up and down among nodes associated with these two agents. From the customer agent' perspective, the updated information contains supplier's own estimation on commitment processing. To some degree, it reflects supplier current ability and attitude of service providing. It is much fresher and convincing than the one that is initially stored in customer agent. Then, the customer agent can calculate the new production of link strength and incoming agent's belief so that a better service provider may be found. This procedure adjusts agent local view through interactions. For example, if the current supplier, say agent A_k's current estimation on commitment processing is not met agent A_i expectation, e.g. the new value of updated e_{ki} and $BEL(x_k = 1)$ is larger than production value of e_{gi} and $BEL(x_g = 1)$. Agent A_i can switch to agent A_g. The following algorithm describes the approach,

Algorithm 5.5:

```
Agent-Switch-Supplier-After-Negotiation
($g_k$, $\{e_{ji}\}$, $\{BEL(x_j = 1)\}$, $A_i^s$, $k$)
Update $e_{ji}$ and $BEL(x_j = 1)$ for all the agents in the set
    $\{A_j \mid g_k = G(A_j), A_j \in A_i^s\}$;
min= $e_{ki} * BEL(x_k = 1)$;
for each $x_j$ that associates with agents in the set $\{A_j \mid$
    $g_k = G(A_j), A_j \in A_i^s\}$
        if (min $> e_{ji} * BEL(x_j = 1)$)
            min= $e_{ji} * BEL(x_j = 1)$;
            k=j;
$v_{ki}$ ="set($l_{ki}$ =1)";
Agent-Operations-on-Local-Network
    ($x_i$, $y_{ki}$, $l_{ki}$, $v_{ki}$, $v_k$, $A_k$);
for $A_j$ in $A_i^s$ and $j \neq k$
    Agent-Operations-on-Local-Network
        ($x_i$, $y_{ji}$, $l_{ji}$, $v_{ji}$, $v_j$, $A_j$);
return
```

It has been noticed that, first, when this algorithm is used, the customer agent has to send out a l' type message to the supplier. The whole set of CDBs associated with upper stream network will be updated, which increases the computation complexity and extends additonal time to finish the end order. Second, even though the current supplier agent has worse performance than customer agent originally expected, because of time constraints and other considerations, without knowing the other agent's the newly computed production value of link strength and belief, the customer might not easily give up the current supplier. Because of these two concerns, to avoid high computation cost with a little performance improvement, agents may choose to use Algorithm 5.5 with low frequency.

5.4 Problem 3: weak link discovery

$eBBN_2$ model can not only be used to design algorithms for functional agents to make reasoning over uncertainty, but also can be used for informational agents to generate information to improve system level performance.

In a MASCM, if knowledge about which agent is most likely to fail in OFPs during a VSP formation process, the system can notify its direct supplier and customer agents so that they can find another robust alternative to avoid performance loss in future transactions. The agent that had worse performance, called the *weak link* of a VSC, is also pushed to improve their service quality in order to survive in the system. In this section, we discuss $MASCM_1$ performance optimization through informational agents using $eBBN_2$ to generate public information related to weak links. First, we formally define MASCM performance. A new type of informational agent used to generate system information is also introduced. Second, we design the algorithm to look for a “critical” agent in the system. Third, we give an algorithm to search for a “fragile” direct supplier for end customer agent.

5.4.1 System performance and Statistical Agent (SA)

For a MASCM, the improvement of end customer satisfaction can establish a positive reputation for the whole system. The increasing number of end orders eventually benefits agents at all tiers in the system Therefore, even though different agents could have conflicting targets, the goal to complete end orders is commonly shared, which can be used to measure the system performance that is formally defined as follows.

Definition 5.1. The system performance of a MASCM is defined by ratio of end orders successfully completed to all orders that arrive at the system. The ratio is also called system efficiency. MASCM A performs better than MASCM B if and only if system efficiency of MASCM A is higher than MASCM B.

One critical problem related to MASCM performance is to answer the question "given an end order, which agent is most likely to let the end order incomplete because it fails to resolve the commitment." The answer to the question can optimize the system performance through the broadcasting this statistical outcome to all functional agents. One direct result is to allow suppliers and customers that connect to the weak link pay more attention when they deal with it. For example, give that agent relatively longer time to resolve the commitment. To some degree, this broadcasting action indirectly fixes the easy-broken link in a virtual supply chain and causes the end order failure possibility to decrease. Analyzing existing solutions to different end orders can generate this type of information. If one agent always performs poorly, e.g. always has the highest possibility to fail in a complete VSC, it can be thought as the weak link.

To setup weak link discovery mechanism, we introduce a new type of informational agent, called Statistical Agent (SA). This agent knows the system structure. It will collect information about a complete VSC for each end order. After analyzing the data, it also takes charge of the system level information broadcasting. However, SA is not involved in any OFPs. Like all other informational agents such as broker, Name Server and so on, it is altruistic and neutral to any parties in the system.

For a system of $MASCM_1$, SA knows the corresponding $eBBN_2$ model structure as well. Of course, the detailed knowledge that SA keeps may be out of date and inaccurate if some agents reluctant to post their information to the outside. However, the information provide by SA is a good reference for agents to determine weak links. In the following sections, we discuss alogritms used by SA for weak link discovery based on $eBBN_2$ model.

5.4.2 Critical agent detection

To the end customer agent, other agents in the system are its direct or indirect suppliers. One failure in these agents might lead to its end-order processing failure. The critical agent is defined as the agent in the system that is most likely cause the end order to fail in an existing solution. The formal definition is as follows,

Definition 5.2. A critical agent A_i in a complete VSC is the one associate with node x_i that $\forall x_j \in eBBN_2$ and $j \neq i$, $p(x_1 = 1 \mid x_j = 1) \leq p(x_1 = 1 \mid x_i = 1)$.

SA can search for the “most” critical agent in the system using the following algorithm,

Algorithm 5.6:

```
Critical-Agent-Detection ([$n_k$], {$VSC_i$}, $A_0$)
for each $n_k$ in [$n_k$]
    $n_k = 0$;
for each $VSC_i$ in {$VSC_i$}
    max=0; k=-1;
    for each $A_j$ in $VSC_i$
        $x_j$ =1
        Based on Equation (5.1) -(5.6) generate $\Pi(\lambda)$ messages
        Update the probability distribution of nodes in $VSC_i$
        Compute $BEL(x_1 = 1)$
        if ( $BEL(x_1 = 1)$ >max)
            max= $BEL(x_1 = 1)$;
            k=j;
        reset $VSC_i$;
    $n_k = n_k + 1$;
search maximum $n_k$
return $A_k$
```

In the algorithm, no agent actions are involved. The model of $eBBN_2$ that SA uses to analyze the data is equivalent to the model of $eBBN_1$. The probability propagation follows Equation (5.1) till (5.6).

The size of array $[n_k]$ is equal to the number of agents in the system. The set $\{VSC_i\}$ is used to store the solutions to different end orders coming to the system. Since we only compare the effect from one supplier to the end customer agent, during the whole procedure only the type of Π messages are necessary to be generated and sent to the others. If there are m functional agents in the system, time complexity of the first loop is $O(m)$; the second loop is $O(m^2 n)$ considering that there are n different solution SA calculate and it takes $O(m)$ to update the probability distribution in the most inner loop (a complete VSC_i can be represented as a tree followed Proposition 4.3); the search procedure time complexity is $O(m)$ using any standard searching algorithm. Thus, the time complexity of Algorithm 5.6 is $O(m^2 n)$. The space complexity is $O(m,n)$.

5.4.3 Fragile agent detection

Algorithm 5.6 discussed in last section give the answer of the weak link to end customer in a top-down manner (from supplier to customer). The question can be solved in a reversed way, that is, if the customer order fails, which agent should be responsible for that. Or in the model $eBBN_2$, when child node probability changes, which one of its ancestor has the largest failure possibility? We define the fragile agent in a complete VSC is the agent that has maximum belief on commitment failure when the end customer agent cancels its order (node x_1 is instantiated as 1). The formal definition is given as follows,

Definition 5.3. A fragile agent A_i in a complete VSC is the one associate with node x_i that $\forall x_j \in eBBN_2$ and $j \neq i$, $p(x_j = 1 \mid x_1 = 1) \leq p(x_i = 1 \mid x_i = 1)$.

The SA can use following algorithm to determine the most fragile agent among a group of existing solutions.

Algorithm 5.7:

```
Fragile-Agent-Detection (([$n_k$], {$VSC_i$}, $A_0$)
for each $VSC_i$ in {$VSC_i$}
        max=0; k=-1;
        $x_0$=1;
        Based on Equation (5.1)- (5.6) generate $\Pi(\lambda)$ messages;
        Update the probability distribution of nodes in $VSC_i$;
        for each $A_j$ in $VSC_i$
                if ($BEL(x_j = 1)$ >max)
                max= $BEL(x_j = 1)$;
                k=j;
                reset $VSC_i$;
$n_k = n_k + 1$;
search maximum $n_k$
return $A_k$
```

Similar to Algorithm 5.6, in the system it is only necessary to generate and propagate the type of l messages since we only consider the decision affection from the customer to suppliers. Algorithm 5.7 has the time and space complexity as the algorithm 5.4. However, given the same set of complete VSCs they can return different results to SA.

5.5 An example

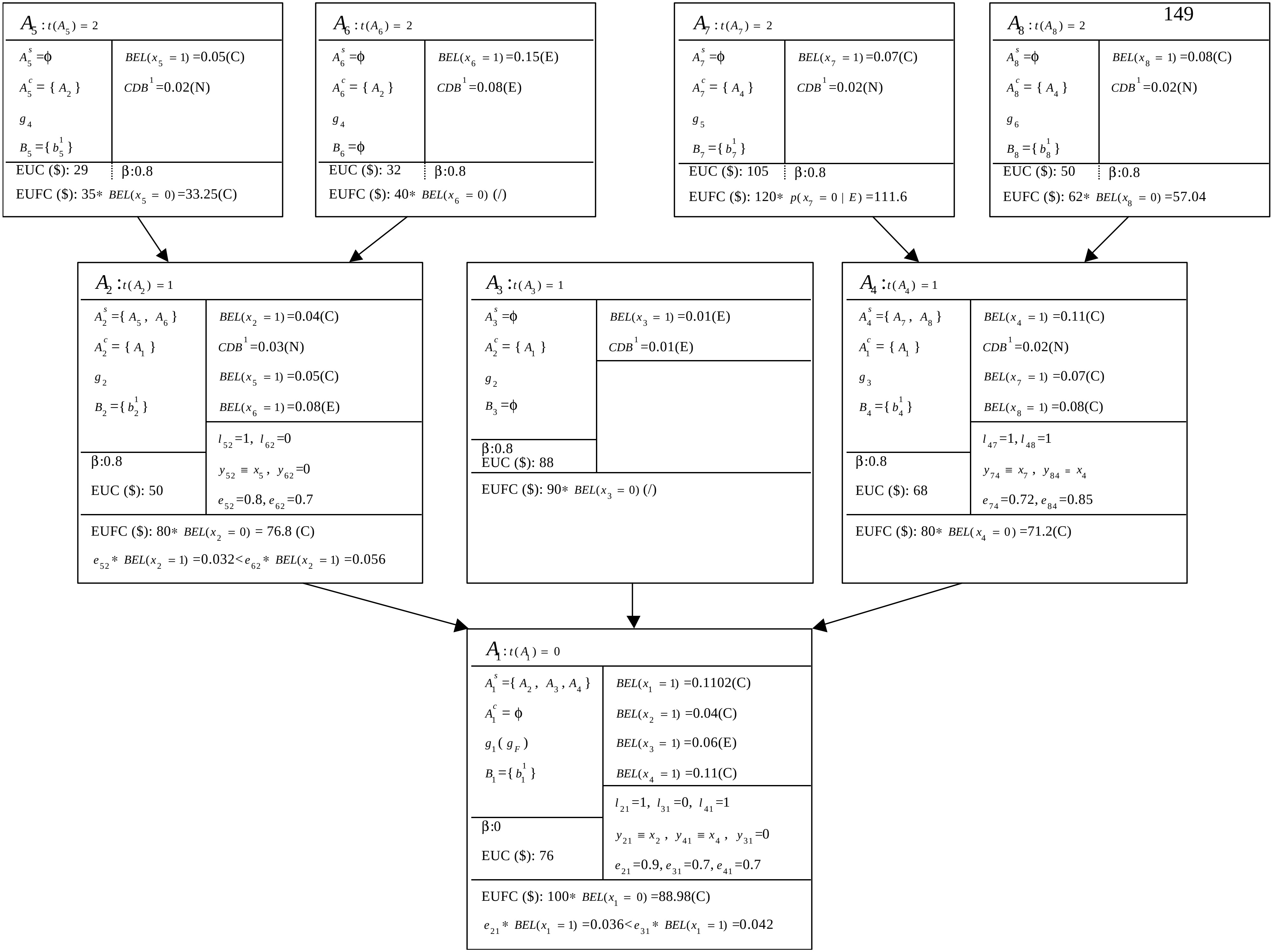
In this section, we show two consecutive states of a small $MASCM_1$, which consists of eight functional agents. Agents use algorithms that we discussed above to retrieve, broadcast and reason the impact of uncertain events.

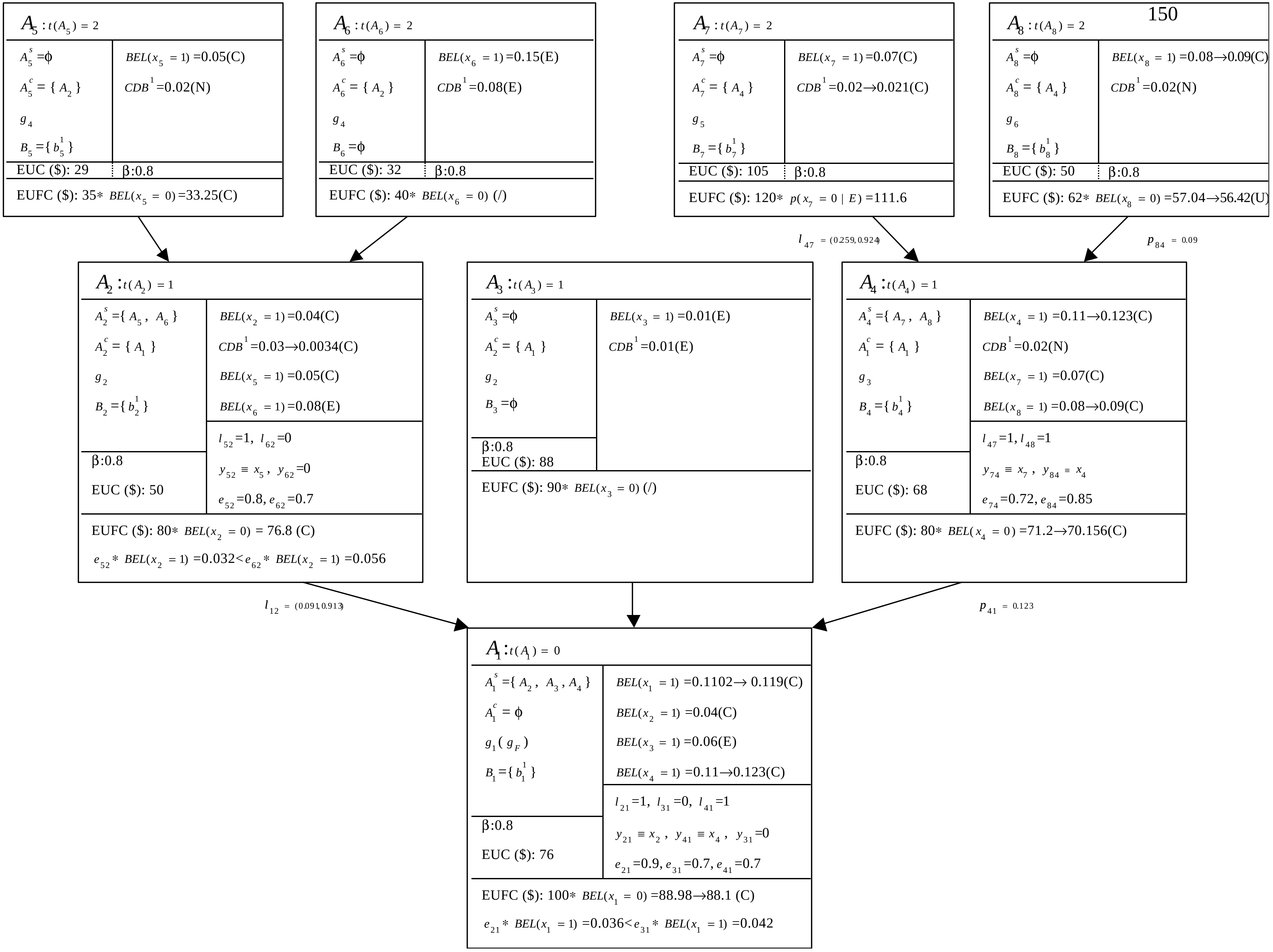
In the sample system, corresponding to each steps of OFP, each agent implements Algorithm 5.1 to Algorithm 5.7 to deal with uncertainty. In other words, it follows Algorithm 5.4 to select a direct supplier when it starts an OFP; After its negotiation process, it will follows Algorithm 5.5 to switch suppliers; it uses Algorithm 5.2 and 5.3 to decide whether it needs to cancel the commitment to the customer when it retrievers information from $eBBN_2$ units associated with upstream and downstream respectively.

In the example, we simplify the EUFC as the production of expected profit and agent's belief on commitment success. In the system, $l^{'}(\Pi)$ messages just contain a single (pair of) real number. This number is directly used Equation (5.7), (5.8) and (5.12) without further modification.

We use two figures similar to SDG defined in last chapter to describe the agent system states. An agent and $eBBN_2$ unit associated with it are represented as a node in the graph. The directed link represents the supplier-customer relationship. In a node, we list the basic attributes of an agent and corresponding $eBBN_2$ components. For attributes that change over time, different labels are attached as timestamps. The label (C) indicates the value is current; (E) indicates the attribute has initial value. An agent sets the value according to its estimation; (N) indicates the value is reached at the negotiation period; (/) indicates the value is not available at the given time point. In order to show the value change, a directed link is directed from the old one to the current one.

In the first graph, there is an evolving VSC in the system. It consists of agent A_1, A_2, A_4, A_5, A_7 and A_8. None of these agents have completed their on-hold commitments so far. In the second graph, it shows the system state change and $eBBN_2$ update when agent A_8 observed an unexpected event. The graph shows changes in the decision procedure as well as ones in corresponding $eBBN_2$ model.


$A_5 : t(A_5) = 2$
$A_5^s = \phi$
$A_5^c = \{ A_2 \}$
g_4
$B_5 = \{ b_5^1 \}$
$BEL(x_5 = 1) = 0.05$(C)
$CDB^1 = 0.02$(N)
EUC ($): 29
β:0.8
EUFC ($): 35* $BEL(x_5 = 0) = 33.25$(C)
$A_6 : t(A_6) = 2$
$A_6^s = \phi$
$A_6^c = \{ A_2 \}$
g_4
$B_6 = \phi$
$BEL(x_6 = 1) = 0.15$(E)
$CDB^1 = 0.08$(E)
EUC ($): 32
β:0.8
EUFC ($): 40* $BEL(x_6 = 0)$ (/)
$A_7 : t(A_7) = 2$
$A_7^s = \phi$
$A_7^c = \{ A_4 \}$
g_5
$B_7 = \{ b_7^1 \}$
$BEL(x_7 = 1) = 0.07$(C)
$CDB^1 = 0.02$(N)
EUC ($): 105
β:0.8
EUFC ($): 120* $p(x_7 = 0 \mid E) = 111.6$
$A_8 : t(A_8) = 2$
$A_8^s = \phi$
$A_8^c = \{ A_4 \}$
g_6
$B_8 = \{ b_8^1 \}$
$BEL(x_8 = 1) = 0.08$(C)
$CDB^1 = 0.02$(N)
EUC ($): 50
β:0.8
EUFC ($): 62* $BEL(x_8 = 0) = 57.04$
$A_2 : t(A_2) = 1$
$A_2^s = \{ A_5 , A_6 \}$
$A_2^c = \{ A_1 \}$
g_2
$B_2 = \{ b_2^1 \}$
$BEL(x_2 = 1) = 0.04$(C)
$CDB^1 = 0.03$(N)
$BEL(x_5 = 1) = 0.05$(C)
$BEL(x_6 = 1) = 0.08$(E)
$l_{52} = 1, \ l_{62} = 0$
$y_{52} \equiv x_5 , \ y_{62} = 0$
$e_{52} = 0.8, e_{62} = 0.7$
β:0.8
EUC ($): 50
EUFC ($): 80* $BEL(x_2 = 0) = 76.8$ (C)
$e_{52} * BEL(x_2 = 1) = 0.032 < e_{62} * BEL(x_2 = 1) = 0.056$
$A_3 : t(A_3) = 1$
$A_3^s = \phi$
$A_2^c = \{ A_1 \}$
g_2
$B_3 = \phi$
$BEL(x_3 = 1) = 0.01$(E)
$CDB^1 = 0.01$(E)
β:0.8
EUC ($): 88
EUFC ($): 90* $BEL(x_3 = 0)$ (/)
$A_4 : t(A_4) = 1$
$A_4^s = \{ A_7 , A_8 \}$
$A_1^c = \{ A_1 \}$
g_3
$B_4 = \{ b_4^1 \}$
$BEL(x_4 = 1) = 0.11$(C)
$CDB^1 = 0.02$(N)
$BEL(x_7 = 1) = 0.07$(C)
$BEL(x_8 = 1) = 0.08$(C)
$l_{47} = 1, l_{48} = 1$
$y_{74} \equiv x_7 , \ y_{84} \equiv x_4$
$e_{74} = 0.72, e_{84} = 0.85$
β:0.8
EUC ($): 68
EUFC ($): 80* $BEL(x_4 = 0) = 71.2$(C)
$A_1 : t(A_1) = 0$
$A_1^s = \{ A_2 , A_3 , A_4 \}$
$A_1^c = \phi$
$g_1 (g_F)$
$B_1 = \{ b_1^1 \}$
$BEL(x_1 = 1) = 0.1102$(C)
$BEL(x_2 = 1) = 0.04$(C)
$BEL(x_3 = 1) = 0.06$(E)
$BEL(x_4 = 1) = 0.11$(C)
$l_{21} = 1, \ l_{31} = 0, \ l_{41} = 1$
$y_{21} \equiv x_2 , \ y_{41} \equiv x_4 , \ y_{31} = 0$
$e_{21} = 0.9, e_{31} = 0.7, e_{41} = 0.7$
β:0
EUC ($): 76
EUFC ($): 100* $BEL(x_1 = 0) = 88.98$(C)
$e_{21} * BEL(x_1 = 1) = 0.036 < e_{31} * BEL(x_1 = 1) = 0.042$



$A_5 : t(A_5) = 2$

$A_5^s = \phi$

$A_5^c = \{ A_2 \}$

g_4

$B_5 = \{ b_5^1 \}$

$BEL(x_5 = 1) = 0.05$(C)

$CDB^1 = 0.02$(N)

EUC ($): 29 β:0.8

EUFC ($): 35* $BEL(x_5 = 0) = 33.25$(C)

$A_6 : t(A_6) = 2$

$A_6^s = \phi$

$A_6^c = \{ A_2 \}$

g_4

$B_6 = \phi$

$BEL(x_6 = 1) = 0.15$(E)

$CDB^1 = 0.08$(E)

EUC ($): 32 β:0.8

EUFC ($): 40* $BEL(x_6 = 0)$ (/)

$A_7 : t(A_7) = 2$

$A_7^s = \phi$

$A_7^c = \{ A_4 \}$

g_5

$B_7 = \{ b_7^1 \}$

$BEL(x_7 = 1) = 0.07$(C)

$CDB^1 = 0.02 \rightarrow 0.021$(C)

EUC ($): 105 β:0.8

EUFC ($): 120* $p(x_7 = 0 \mid E) = 111.6$

$A_8 : t(A_8) = 2$

$A_8^s = \phi$

$A_8^c = \{ A_4 \}$

g_6

$B_8 = \{ b_8^1 \}$

$BEL(x_8 = 1) = 0.08 \rightarrow 0.09$(C)

$CDB^1 = 0.02$(N)

EUC ($): 50 β:0.8

EUFC ($): 62* $BEL(x_8 = 0) = 57.04 \rightarrow 56.42$(U)

$l_{47} = (0.259, 0.924)$

$p_{84} = 0.09$

$A_2 : t(A_2) = 1$

$A_2^s = \{ A_5, A_6 \}$

$A_2^c = \{ A_1 \}$

g_2

$B_2 = \{ b_2^1 \}$

β:0.8

EUC ($): 50

$BEL(x_2 = 1) = 0.04$(C)

$CDB^1 = 0.03 \rightarrow 0.0034$(C)

$BEL(x_5 = 1) = 0.05$(C)

$BEL(x_6 = 1) = 0.08$(E)

$l_{52} = 1$, $l_{62} = 0$

$y_{52} \equiv x_5$, $y_{62} = 0$

$e_{52} = 0.8$, $e_{62} = 0.7$

EUFC ($): 80* $BEL(x_2 = 0) = 76.8$ (C)

$e_{52} * BEL(x_2 = 1) = 0.032 < e_{62} * BEL(x_2 = 1) = 0.056$

$A_3 : t(A_3) = 1$

$A_3^s = \phi$

$A_2^c = \{ A_1 \}$

g_2

$B_3 = \phi$

β:0.8

EUC ($): 88

$BEL(x_3 = 1) = 0.01$(E)

$CDB^1 = 0.01$(E)

EUFC ($): 90* $BEL(x_3 = 0)$ (/)

$A_4 : t(A_4) = 1$

$A_4^s = \{ A_7, A_8 \}$

$A_1^c = \{ A_1 \}$

g_3

$B_4 = \{ b_4^1 \}$

β:0.8

EUC ($): 68

$BEL(x_4 = 1) = 0.11 \rightarrow 0.123$(C)

$CDB^1 = 0.02$(N)

$BEL(x_7 = 1) = 0.07$(C)

$BEL(x_8 = 1) = 0.08 \rightarrow 0.09$(C)

$l_{47} = 1$, $l_{48} = 1$

$y_{74} \equiv x_7$, $y_{84} \equiv x_4$

$e_{74} = 0.72$, $e_{84} = 0.85$

EUFC ($): 80* $BEL(x_4 = 0) = 71.2 \rightarrow 70.156$(C)

$l_{12} = (0.091, 0.913)$

$p_{41} = 0.123$

$A_1 : t(A_1) = 0$

$A_1^s = \{ A_2, A_3, A_4 \}$

$A_1^c = \phi$

$g_1 (g_F)$

$B_1 = \{ b_1^1 \}$

β:0.8

EUC ($): 76

$BEL(x_1 = 1) = 0.1102 \rightarrow 0.119$(C)

$BEL(x_2 = 1) = 0.04$(C)

$BEL(x_3 = 1) = 0.06$(E)

$BEL(x_4 = 1) = 0.11 \rightarrow 0.123$(C)

$l_{21} = 1$, $l_{31} = 0$, $l_{41} = 1$

$y_{21} \equiv x_2$, $y_{41} \equiv x_4$, $y_{31} = 0$

$e_{21} = 0.9$, $e_{31} = 0.7$, $e_{41} = 0.7$

EUFC ($): 100* $BEL(x_1 = 0) = 88.98 \rightarrow 88.1$ (C)

$e_{21} * BEL(x_1 = 1) = 0.036 < e_{31} * BEL(x_1 = 1) = 0.042$

5.6 Advantages of using eBBN approach in algorithm design

In this section, we discuss the advantage to use eBBN approach to design algorithms related to agent uncertain management.

First, $eBBN_2$ provides a quantitative approach to design algorithms that agent can use to make reasoning over uncertainty, e.g. measure the impact of uncertain factors on the internal commitment processing. Using eBBN approach lets algorithm design have an objective and scientific basis instead of subjective human being's experience. It also makes the software agent become less dependent on the human interaction.

Second, eBBN model only requires local computation and message content are simple, so that algorithms design require less global knowledge. For example, from algorithm 5.5 we can see that each agent only uses locally stored information such as link strength and initial agent belief to calculate its belief update. The propagated message $\Pi(l^{'})$ can be as simple as $p(x_i = 1 | E)$. This feature let $eBBN_2$ become a feasible solution to implement uncertainty mechanism for individual functional agent with different owners.

Furthermore, eBBN is an asynchronized and distributed model Thus, algorithm design based on this approach do not enforce when functional agents have to generate and propagate the information, and no central coordinators are required. These mechanisms match MASCM distributed features and marketplace properties.

5.7 Summary

In this chapter, we discuss how to use eBBN model to design algorithms for individual agents to deal with uncertainty. We identify three problems as examples. For each problem, algorithms that utilize information retrieved or stored in the eBBN are presented. An example system, in which each functional agent uses the algorithm we design, is given. We also discuss the advantage of using eBBN approach in uncertainty management design.

Chapter 6 Experiment

eBBN model not only can be used as a theoretical basis to design algorithm for agent uncertainty management, but also can serve as an analytical platform to study the relationship between these mechanisms and overall system performance. In this chapter, we discuss the experiment we made on a small $MASCM_1$. First, we discuss the simulation design and implementation. Then, we present the experiment result and analysis.

6.1 Simulation design and implementation

The goal of our experiment is to study the relationship between individual agent uncertain mechanisms and overall system performance. We compare system performance when agents implement two different sets of rules to update agent beliefs. In one setting, each agent will update its belief exactly follow rules related on $eBBN_2$ operations discussed in Section 5.1. That is, agents immediately update their belief when they observe an unexpected event; in the other setting, agents only update beliefs when commitments are close to resolve. To allow the experiment results are comprisable, two sets of rules are tested in the same $MASCM_1$.

6.1.1 Simulated MASCM and its eBBN model

The $MASCM_1$ that we simulate is the one discussed in Section 5.5. The system consists of eight different functional agents. They sit at three different tiers. At tier 0, there is only one end customer agent. At tier 1, there are three functional agents. They are suppliers of the end customer agents. At tier 2, there are four agents. All of them are raw materials providers. The agents at tier two are their direct customers. No informational agents are in the system. All agents have known their direct customer and suppliers at system design time. When there is an end order arrives in the system, the inter-connected OFP will be triggered and an evolving VSC emerges. The overall system performance is defined as the Definition 5.1.

An $eBBN_2$ model is established for this $MASCM_1$ following our discussion in Chapter 4. In the model, agents' strategic behaviors that de-commit their previous agreements or choose different suppliers are two specified sources that instantiate random variables according to rules we discuss in Section 4.7. Other unexpected internal or external events are treat as random stimuli that cause probability distribution of commitment failure variables to change.

Each agent A_i controls a basic unit of $MASCM_1$ defined in Section 5.1. Their chain related behaviors exactly follow the definition of OFP in Chapter 2.

6.1.2 Implementation details

The simulated system is implemented in Java 1.3SE. Each agent has similar architecture (see Section 5.5), and is implemented as an object. All agents run in the same Java virtual machine. Their initial parameters, for example, suppliers and customers are stored in the property files.

In the system if an agent is not a raw material provider, its behaviors can be described as following procedure.

Procedure 6.1.

```
Randomly adjust its internal parameters including CT, EUFC and $\beta$
If receiving an order from the its customer,
    Then make a commitment to it.
Select direct suppliers using Algorithm 5.4 for all products it needs
    Send an order to each of selected agents
Wait for the commitment processing message from its direct suppliers for all ongoing OFPs
    If receiving a $\Pi$ message
        Then
        Make decision on Commitment Cancellation using Algorithm 5.2;
        Make decision on Supplier Switch using Algorithm 5.5;
        Operate local unit of $eBBN_2$ using Algorithm 5.1;
    If receiving a $\lambda$ message
        Then
        Make decision on Commitment Cancellation using Algorithm 5.3;
```

In the simulated system unexpected events occur in raw material providers. It is represented as a random variable between 0 and 1 (Interval (0,1)). If the event happens to an active raw material agent, the value is recorded for updating agent's belief on commitment failure. This update may affect the whole system performance through sell-buy connections. We define two different sets of rules for raw material agents to update its beliefs. They are listed as below.

Rule set 1.

If there is unexpected event occurring in an active raw material agent, it will update its agent belief immediately and propagate down to direct customers.

Rule set 2.

If there is an unexpected event occurring in an active raw material agent, the possible change of agent belief on commitment failure is temporally stored. When it is the time to resolve the commitment, if this agent is still active, it uses a specific procedure to calculate the accumulated impacts of uncertain events on its agent belief then propagate it down to direct customers.

It is obvious Rule 1 exactly follows our discussion on $eBBN_2$ unit operations discussed in Section 5.1. Two things need to notice related to Rule 2. First, when Rule 2 is used for raw material agent to generate and propagate information, the agent actually sends out its final guess. Thus other agents in the system only have limited times to select a direct supplier for a group of agents that provide the same product following Algorithm

5.5 (Procedure 6.1 line 8). In other words, if these agents choose agent A at its direct supplier at the beginning then switch to agent B, it can not reselect agent A again later. Second, each agent has a special procedure to calculate the total impacts of unexpected events in an active agent. It is as follows.

Procedure 6.2.

Unexpected Event is denoted as UE, which represented as a random number as we discuss before. Agent temporal belief is denoted as ATB, which represents the trend of commitment processing when an unexpected event occurs. If UE is larger than ATB, raw material providers will guess the commitment is likely to failure than before. If UE is smaller than ATB, they will guess the commitment has more chance to resolve.

```
UE>ATB,
        n=0;
        while ((ATB+UE/$2^n$)>1)
                UE=UE/$2^n$;
                n=n +1;
        ATB = ATB+UE;
UE<ATB,
        n=0;
        while ((ATB-UE/$2^n$)<0)
                UE=UE/$2^n$;
                n=n +1;
        ATB = ATB-UE;
```

Followed the discussion above, raw material agents' behaviors are described as follows,

Procedure 6.3.

```
Randomly adjust its internal parameters including CT, EUFC and $\beta$
If receiving an order from the its customer,
        Then make a commitment to it.
If it is active and there is unexpected event occurring in it
        Then
        Update agents beliefs follow Rule 1 or Rule2 (Procedure 6.2)
                Make decision on Commitment Cancellation using Algorithm 5.2;
                Operate local unit of $eBBN_2$ using Algorithm 5.1;
        If receiving a $\lambda$ message
                Then
                Make decision on Commitment Cancellation using Algorithm 5.3;
```

At initial time, all agents in system are inactive. When an end order arrives, it causes the end customer agent become active and generates orders to selected direct suppliers. These agents become active. An evolving VSC emerges. During end order solving process, certain numbers of unexpected events occur in raw material agents. The effects of these unexpected events propagate through sell-buy connections, which may or may not eventually cause the end customer agent to flip from active to inactive. If, after these unexpected events happen, the end customer agent remains active status, we say the end order is solved successfully. If it is become inactive, the system fails to find a solution to this end order. No matter in which case, the system is reset to initial state. The end customer agent becomes inactive. An end order is continuously generated to the system whenever the end customer agent is inactive.

The following table summarizes the difference between two settings of simulation.

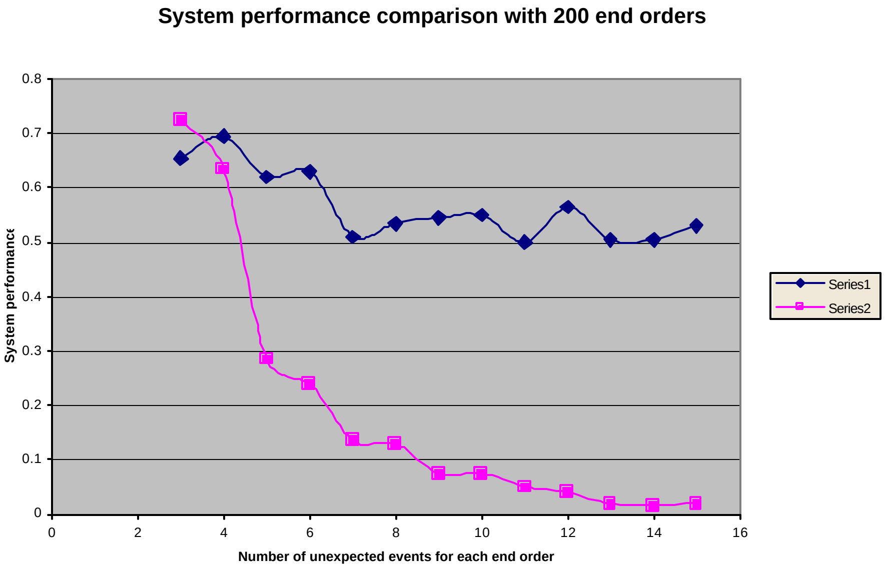
	Simulation Setting 1	**Simulation Setting 2**
Uncertainty mechanism used in raw material providers	Rule set 1; exactly following discussion in Section 5.1	Rule set 2; Procedure 6.2.
Uncertainty mechanism used in other agents	Unlimited times to select direct suppliers from a group of agents that provide the same product.	Limited times to select direct suppliers from a group of agents that provide the same product.

Table 6.1 The difference between two simulation settings

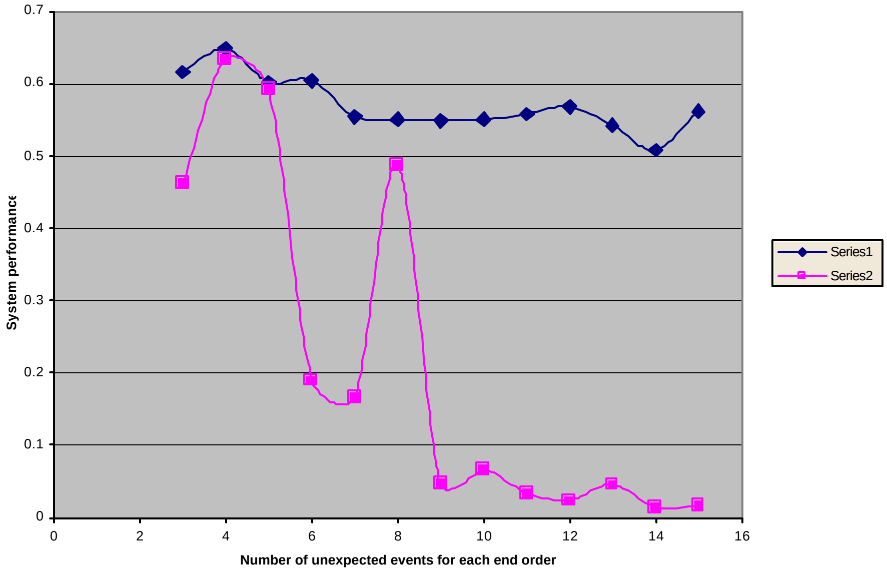
6.2 Experiment result and analysis

In this section we present experiment result and analysis. In our experiment, for each simulation setting, we continuously generate certain number of end orders to the system. During each end order resolving process, certain unexpected events randomly occur to raw material providers. We compare their system performance.

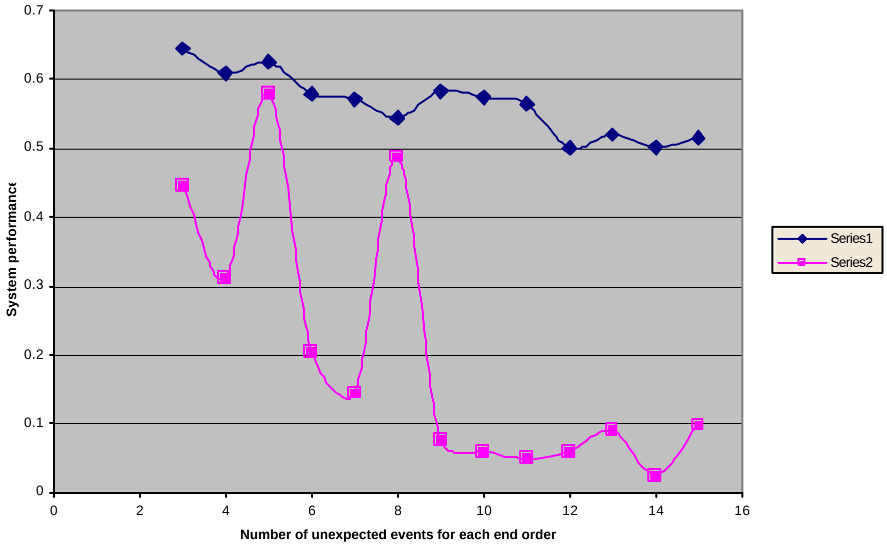
We have tested cases when there are 200, 500, 800,1000, 1600,2200 and 3000 continuous end orders to the system, and during each end order solving process there are 3 to 15 times of unexpected events occurring in raw material providers. Each of the following figures shows an indivdual testing result with different numbers of end orders. Each data point represents ratio of the number of successful completed end orders to the various number of end orders (Y axis) under different number of uncertain events (X axis).



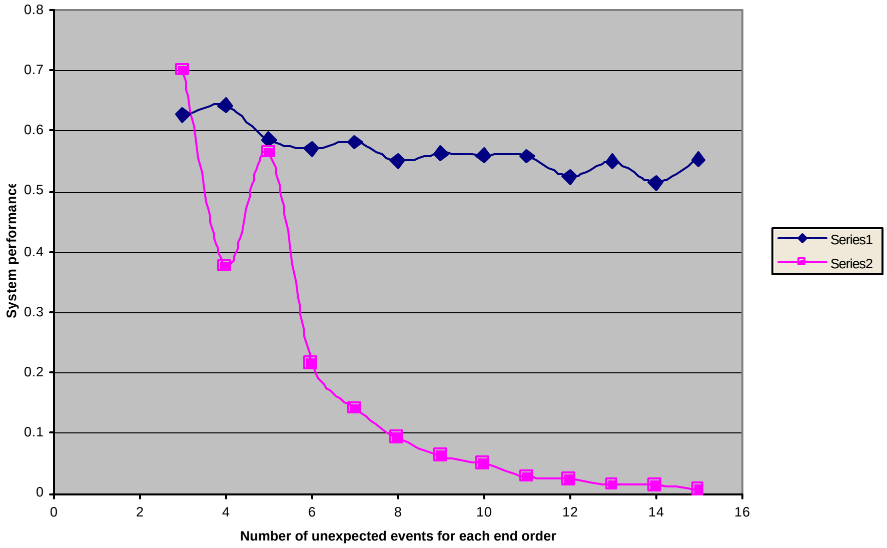
System performance comparison with 500 end orders



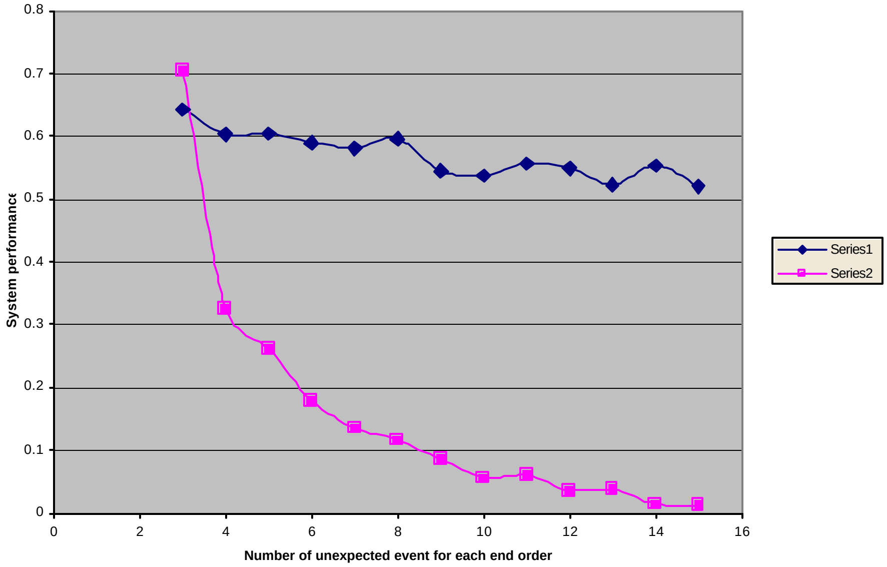
System performance with 800 end orders



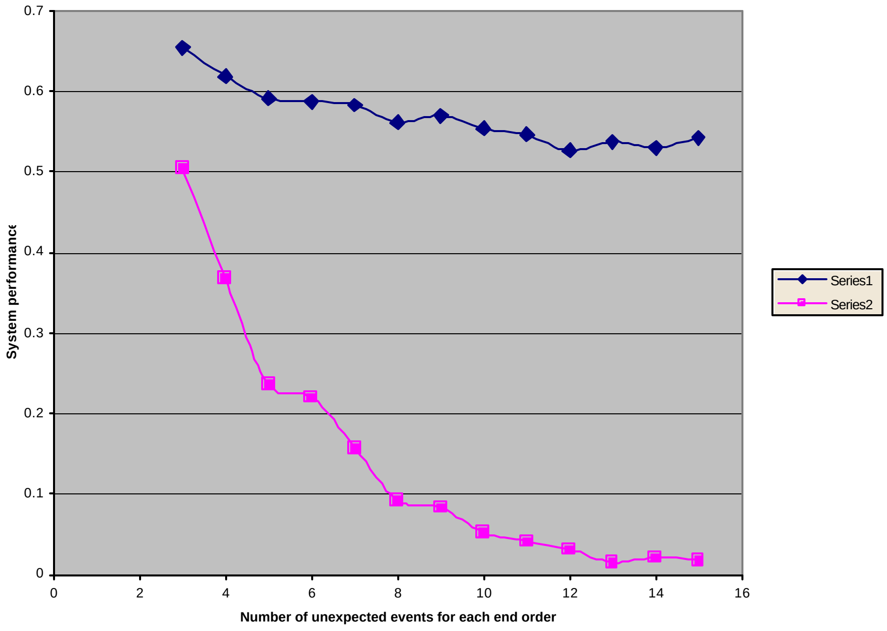
System performance with 1000 end orders



System performance comparison with 1600 end orders



System performance comparison with 2200 end orders



System performance comparison with 3000 end orders

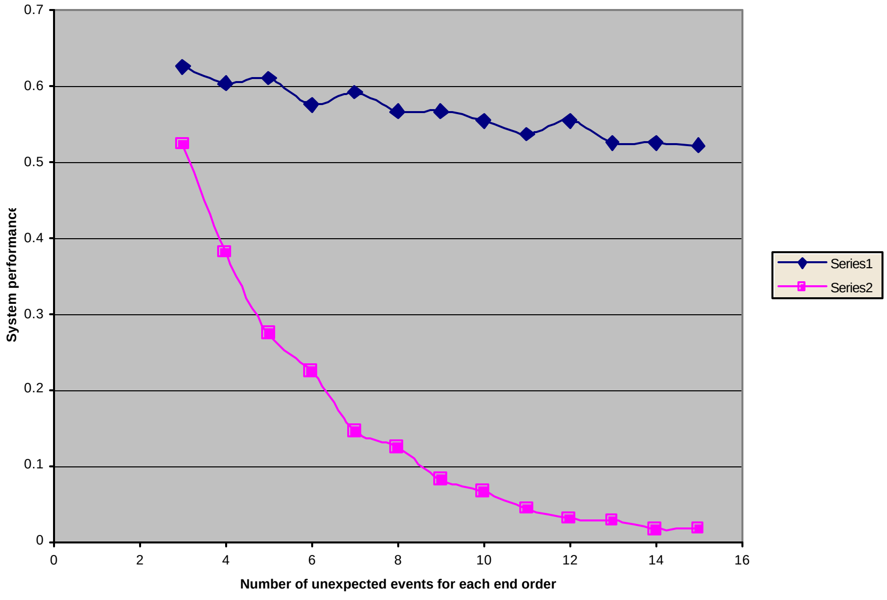


Figure 6.1 Experiment result

(Note: series1 is the experiment under simulation setting 1; on the contrary, seriers2 is the experiment under simulation setting2.)

Form the Figure 6.1 we can see if other conditions keep same, system has more stable performance in simulation setting 1 than in simulation setting 2. It can be explained that in setting 1, when an unexpected event occurs in an active raw material provider, the components of an evolving VSC are automatically update according to this change. The impact of this event has been clean up right away if possible. Therefore, although when uncertain events occur in a high frequency, the system performance changes little. On the contrary in setting 2, the impact of uncertainty is, to some degree, accumulated, which has more chance to move agent beliefs toward the direction that eventually causes the agent or its downstream agents to cancel commitments. In addition, since in setting two agents only have limited times to choose a direct supplier from a group of agent that provides the same products, which further restricts the adjustment capability of an evolving VSC facing to unexpected challenges. Thus, when uncertain events appear often, the system performance goes down dramatically.

6.3 Summary

In this Chapter we discuss the experiment we made on a small MASCM. In the experiment, eBBN is used as an analytic platform to study the relationship between uncertainty mechanism and overall system performance in two different setting. We find when agent use the uncertainty mechanism that exactly follows our discussion in Chapter 5 has more stable performance than the competitor.

Chapter 7 Conclusion and future research directions

In this chapter we summarize this dissertation research and discuss future research directions.

7.1 Achievements

In this dissertation we presented a theoretical model, called extended Bayesian Belief Network (eBBN), to formalize functional agent interaction in the uncertain environment. The impact of an observed unexpected event is formalized as agent's belief on commitment failure; the causal relationship related to the agent's belief exhibiting in the system is model as links; agents' effort to share information and knowledge related to uncertainty is generalized as agents' belief update and propagation over links.

eBBN model extends the representation and inference capability of traditional Bayesian Belief networks (BBNs). It innovatively introduces both Y type and L type nodes to catch the change of demand-supply relationship so that, unlike other BBNs, it can describe the dynamically updated causal relationship in a complex agent system with the evolvement of a VSC. Based on this representation a set of equations used for agents to reason over uncertain factors are deduced. These equations are easily used in agent decision procedures. Through properly extending previous works and defining the concept of actions, eBBN can study the effect of agents' strategic behaviors on other agents' belief change.

We specifically study a serial eBBN models, $eBBN_0$, $eBBN_1$ and $eBBN_2$ for a simple MASCM, called $MASCM_1$. The model of $eBBN_2$ is further used as a basis to design algorithms for uncertainty management of an individual functional agent, including Commitment Cancellation and Supplier Selection and Switch. In the system, if there exists a type of informational agent, called Statistical Agents (SA), this type of agents can use this model to generate public information on Weak Link Discovery, which can help improve overall system performance.

We use an $eBBN_2$ model as an analytical platform to study the relationship between uncertainty mechanisms of individual agents and overall system performance. We find that when agents exactly follow algorithms of uncertainty management and operations on the local unit of the model that we present, the overall system performance is stable.

7.2 Model limitations and future study

In this dissertation, we do not present a general model for any MASCM. The most sophisticated eBBN model we developed is $eBBN_2$, which only formalizes interaction between functional agents when each of them holds one customer commitment at one time. But in many practical scenarios, agent behaviors might not follow this assumption. For example, a customer agent might make two different orders at the same time. This requires that we further extend $eBBN_2$ model.

When we limit the number of commitments a functional agent holds, $eBBN_2$ actually models one slice of multi-threaded activities occurring among functional agents. Each of these slices can be thought as a $MASCM_1$. Thus, one solution to studying agent interactions in complex MASCM is to represent its states as multiple but connected DSDC graphs, each of which corresponds to a $eBBN_2$ graph. For example if involved in two OFPs triggered by its direct customers, functional agent A_i manipulates two sets of nodes, each of which is part of individual $eBBN_2$ model. Agent A_i uses internal business logics to control the connection between these different sets of variables. In other words, we use two or more $eBBN_2$ graphs to formalize agent interactions in a complex MASCM. The solution is feasible and does not require more on theoretical exploration. But it requires designing an additional mechanism for agent to coordinate multiple $eBBN_2$ models.

The other possible solution is to extend the definition of node (hype-node) and link (hype-link). Each of them can have complicated internal structures. In other words, a hype-node is not equal to a random variable anymore but consists of connected variables with the same type, e.g. two commitment failure variables. A hype-link represents the causal relationship between two hype-nodes rather than the one between two random variables. In this way, a MASCM can be still represented as a single $eBBN_2$ graph. Agents follow the simple operations defined in Chapter 5. But agent interactions may cause a relatively complex change of hype-nodes. This solution shifts the difficulty from the agent design to BBN concept clarification. It needs more elaboration and fundamental research.

Based on the discussion above, we hope that the progress achieved in this research will provide a substantial understanding of agent interactions in an uncertain environment and will serve as the fundation for studying the relationship between local decisions and global performance.

Bibliography

[1] N.Gregory. Mankiw, *Principles of Microeconomics*, Dryden Press, Fort Worth 1998.

[2] H. L. Lee, V. Padmanabhan, and S. Whang, “The Bullwhip Effect in Supply Chains,” *Sloan Management Reviews*, Spring, 1997

[3] M. Barbuceanu, and M. S. Fox, “The Information Agent: An Infrastructure Agent Supporting Collaborative Enterprise Architectures,” in *Proceedings of Third Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, Morgantown, West Virginia, IEEE Computer Society Press, 1994.

[4] R. B. Handfield, and Ernest L. Nichols, Jr., *Introduction to Supply Chain Management*, Prentice Hall, New Jersey, 1999.

[5] H. S. Nwana, “Software Agent*:* Overview,” *Knowledge Engineer Review*, Vol.11, No 3, pp. 205-244, October/November 1996. Cambridge University Press, 1996.

[6] Y. Chen et al., “A Negotiation-based Multi-agent System for Supply Chain Management,” in *worknote of Autonomous Agents’99 workshop on Agent-based Decision-support for Managing the Internet-enabled supply-chain*, Seattle, Washington State, U.S, May 1, 1999.

[7] J. Odell, and C. Bock, “Suggested UML Extensions for Agents,” response to the OMG Analysis and Design Task Force UML RTF 2.0 Request for Information, 1999.

[8] B. Bauer, J. P. Müller, and J. Odell **“**Agent UML: A Formalism for Specifying Multiagent Interaction,” *Agent-Oriented Software Engineering*, Springer-Verlag, Berlin, pp. 91-103, 2001.

[9] J. Odell, H. Van Dyke Parunak, and B. Bauer “Extending UML for Agents,”in *Proceeding of the Agent-Oriented Information Systems Workshop at the 17th National conference on Artificial Intelligence*, Gerd Wagner, Austin, TX, pp. 3-17, 2000.

[10] K. Jensen, “Coloured petrit nets and the invariant method,” *Theoretical Computer Science* 14:317-336, 1981.

[11] M. Rusinkiewicz, and A. Seth, “Specification and Execution of Transactional Workflows,” *Modern Database System*, ACM Press and Addison Wesley Publishing Company, pp.592-620, 1995

[12] D. Georgakopoulos, and M. Hornick, “An overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure,” *Distributed and Parallel Databases*, Vol. 3, No. 2, pp. 119-153, 1992.

[13] C. Billington, “Strategic Supply Chain Management,” *OR/MS Today*, pp. 20-27, April 1994.

[14] F, Lin *Reengineering the Order Fulfillment Process in Supply Chain Networks: A Mutliagent Information Systems Approach*. Ph.D. Thesis, University of Illinois at Urbana-Champaign, 1996.

[15] NEVEM-workgroup, *Performance Indicators in Logistics,* IFS Publications, Springer-Verlag, 1989.

[16] R. Teigen, *PMC model for Supply Chain Management Intergration*, Master Thesis, University of Toronto, 1997. Performance: A Framework for Logistics Performance.” *The Logistics and Transportation Review* 31, December 1995.

[17] G. Chow, L. E. Henriksson, and T. D. Heaver, “Strategy, Structure, and

[18] P. W. Balsmerier, and W. J. Voision, “Supply Chain Management: A Time-Based Strategy,” *Industrial Management*, pp. 24-27, September-October 1996.

[19] R. S. Kaplan, and D. P. Norton, *The Balanced Scorecard: Translating Strategy into Action,* Harward Business School Press, Boston, 1996.

[20] M. L. Fisher, J.H. Hammond, W. R. Obermeyer, and A. Raman, "Making Supply Meet Demand in an Uncertain World," *Harvard Business Review*, pp. 83-93, May-June 1994.

[21] H. L. Lee, and C. Billington, "Material Management in Decentralized Supply Chains," *Operations Research* Vol. 41, No. 5. pp. 835-847, 1993.

[22] H. L. Lee, C. Billington, and B. Carter "Hewlett-Packard Gains Control of Inventory and Service through Design for Localization," *Interfaces* Vol. 23, No. 4. pp. 1-11,1993.

[23] T. J. Strader, F. Lin, and M. J. Shaw, "Business-to-Business Electronic Commerce and Convergent Assembly Supply Chain Management," *Journey of information Technology*, 14 pp.361-373, 1999.

[24] T. W. Malone, J. Yates, and R. I. Benjamin, "Electronic Markets and Electronic Hierarchies," *Communication of ACM*, Vol. 30, No. 6, 1987.

[25] T. Berners-Lee, R. Cailliau, A. Luotonen, H. F. Nielsen, and A. Secret, "The World-Wide Web," *Communications of the ACM* Vol. 37, No. 8. pp. 76-82, 1994.

[26] C. Holland, G. Lockett, and I. Blackman "Planning for Electronic Data Interchange," *Strategic Management Journal* Vol. 13. pp. 539-550, 1992.

[27] C. D. Snapp, “EDI Aims High for Global Growth,” *Datamation*, March pp. 77-80,1990.

[28] T. W. Malone, and J.F. Rockart “Computers, Networks, and the Corporation,” *Scientific American* vol. 265, no. 3. pp. 128-136, 1991.

[29] R. T. Erik, *Learning XML,* O’Reilly Association, 2000.

[30] P. Stone, and M. Veloso, “Multiagent Systems: A Survey from a Machine Learning Perspective,” *Under review for journal publication*, 1997.

[31] K. S. Decker, “Distributed problem solving: A survey,” *IEEE Transactions on Systems*, Man, and Cybernetics, Vol. 17, pp. 729-740, September 1987.

[32] T. Finin, Y. Labrou, and J. Mayfield, “KQML as an agent communication language,” in *Jeff Bradshaw (Ed.) Software Agents*, MIT Press, Cambridge, 1997.

[33] H. R. Johnston, and M. R. Vitale, “Creating Advantage with Interorganizational Information Systems,” *MIS Quarterly* vol. 12, no. 2. pp. 153-166.

[34] T. J. Strader, F. Lin, and M. J. Shaw, “Simulation of Order Fulfillment in -Divergent Assembly Supply Chains,” *Journal of Artificial Societies and Social Simulation*, Vol. 1, No. 2,1998.

[35] V. R. Lesser, “Multiagent systems: An emerging subdiscipline of AI,” *ACM Computing Surveys*, Vol. 27, pp. 340--342, September 1995.

[36] K. P. Murphy, “A Brief Introduction to Graphical Models and Bayesian Networks,” *http://www.cs.berkeley.edu/~murphyk/Bayes/bayes.html.*

[37] E. Horvitz, “Thinking Ahead: Continual Computation Policies for Allocating Offline and Real-Time Resources,” in *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, IJCAI '99*, pp. 1280-1286, 1999

[38] E. Horvitz, J. Breese, D. Heckerman, D. Hovel, and K. Rommelse, “The Lumiere Project: Bayesian User Modeling for Inferring the Goals and Needs of Software Users,” in *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, July 1998.

[39] J. Stutz, W. Taylor, and P. Cheeseman, “AutoClass C: General Information,” *http://ic-www.arc.nasa.gov/ic/projects/bayes-group/autoclass/autoclass-cprogram.html#AutoClass C.*

[40] R. Neaplitan, *Probabilistic Reasoning in Expert Systems, Theory and Algorith*ms, John Wiley & Son, New York, 1990.

[41] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kauffman Publishers, California, 1988.

[42] Y. Peng, and J. Reggia, *Abductive Inference Model for Diagnostic Problem Solving*, Springer-Verlag, New York, 1990.

[43] R.A. Howard, and J.E. Matheson, “Influence diagrams,” *Readings on the Principles and Applications of Decision Analysis*, Vol. 2. pp. 721-762, Strategic Decision Group, California, 1981.

[44] S. Olmsted *On representing and solving decision problems*, Ph.D dissertation, Dept. of Engineering-Economic Systems, Stanford, University, 1983.

[45] R. D. Shachter, “*Probabilistic inference and influence diagram*,” *Oper. Res*. 36, pp.589-604, 1988.

[46] J. Pearl, “Fusion, Propagation, and Structuring in Belief Networks,” *Artificical Intelligence*, 29, pp.241-288, 1986.

[47] G. Cooper, “Computational complexity of probabilistic inference using Bayesian belief networks (research note),” *Artificical Intelligence* 42, pp393-405, 1990.

[48] P. Dagum, and M. Luby, “Approximately probabilistic reasoning in Bayesian belief networks is NP-hard,” 91, *Artificical Intelligence* pp.141-153, 1993.

[49] J. Peal, “A Probabilistic Calculus of Actions,” in the *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pp. 454-462, Morgan Kaufmann Publishers, California, 1994.

[50] A. Darwiche, and M. Goldszmidt, “Action Networks: A Framework for Reasoning about Actions and Change under Uncertainty,” in the *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann Publishers, California, 1994.

[51] A. Darwiche, and J. Pearl, “Symbolic Causal Networks for Reasoning about Actions and Plans,” Working Notes: AAAI Spring Symposium on Decision-Theoretic Planning, 1994.

[52] R. Davidson, and M. R. Feling. “A Structured, Probabilistic Representation of Action,” in the *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann Publishers, California, 1994.

[53] N. Kushmerick, S. Hanks, and D. Weld "An Algorithm for Probabilistic Planning," *Technical Report 93-0603*, Department of Computer Science and Engineering, University of Washington.

[54] T. Dean, and M. Wellman *Planning and Control*, Morgan Kaufmann, California, 1991.

[55] M. J. Druzdzel, and H. A. Simon, "Causality in Bayesian belief networks," in the *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence*, pp.3-11 Morgan Kaufmann Publishers, California, 1993.

[56] D. Geiger, T. Verma, and J. Pearl, "Identifying independence in Bayesian networks," *Networks*, V.20, No. 5, pp. 507-534, 1990.

[57] D. L. Bayles, *E-Commerce Logistics and E-fulfillment: Delivering the Goods*, Prentice Hall, 2000.

[58] E. Ephrati and J. Rosenschein, "Deriving consensus in multiagent systems", *Artificial Intelligence*, **87**(1-2), pp. 21-74, 1996.

[59] D. Koller and A. Pfeffer, "Representations and Solutions for game-theoretic problems", *Artificial Intelligence*, **94**(1-2), pp. 167-215, 1997.

[60] S.L. Lauritzen and D.J. Spiegelharter, “Local computation with probabilities on graphical structures, and their application to expert systems”, *Journal of the Royal Statistical Society, Series B*, **50**, pp. 157-244, 1988.

[61] D. Pennock and M. Wellman, “Toward a market model for Bayesian inference”, in *Proceedings of UAI-96*, pp. 405-413, 1996.

[62] D. Pennock and M. Wellman, “Representing aggregate belief through the competitive equilibrium of a securities market”, in *Proceedings of UAI-97*, pp. 392-400, 1997.

[63] J. Dix, M. Nanni, and V.S. Subrahmanian, “Probabilistic agent programs”, Computer Science Department Technical Report, CS-TR-4054, University of Maryland, September, 1999.

[64] D. Poole, “The Independent Choice Logic for modeling multiple agents under uncertainty”, *Artificial Intelligence,* **94**(1-2), pp. 7-56, 1997.

[65] Y. Xiang, D. Poole, and M. Beddoes, “Multiply sectioned belief networks and junction forests for large knowledge-based systems”, *Computational Intelligence*, **9**(2), pp. 171-220, 1997.

[66] D. Koller and A. Pfeffer, “Object oriented belief networks”, in *Proceedings of UAI-97*, pp. 302-313, 1997.