2010

Competitive service market: modeling, storage and management

Shuxing Cheng
Iowa State University

Competitive service market: Modeling, storage and management

by

Shuxing Cheng

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Computer Science

Program of Study Committee:
Carl K. Chang, Major Professor
Johnny S.Wong
Samik Basu
Huaiqing Wu
Dan Zhu

Iowa State University

Ames, Iowa

2010

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

In order to capture the business dynamics underlying SOA-based service systems, we propose and formalize the concept of a competitive service market (CSM). A CSM is composed of a set of composite service providers, each managing a collection of atomic service providers. With the help of service composition protocol, composite service providers are able to invoke atomic services and aggregate them into value-added composite services for servicing various types of customers' requests. Centering around the setting of a competitive service market, our research is separated into three parts:

1. Aiming to support the quantitative-based decision processes of different market players, we construct stochastic models to conduct performance analysis at various levels spanning vertically on the structural hierarchy of the service market.

2. In the context of requirements analysis, we classify the concept of service and service instance in terms of their respective functional and non-functional features. Hereafter, we identify the related storage issues and propose a counting Bloom filter-based hybrid storage architecture for the service registry design underlying the service market. A feature-based service discovery protocol is developed to demonstrate the usefulness of this design.

3. The business relationship between different market players are typically framed through the service level agreements (SLAs), which specify the attributes of QoS-based metrics and service costs for the realized service provisioning. SLAs constitute the backbone structure for managing the CSM. We identify several SLA

design patterns in terms of different business scenarios that can occur in the life cycle of a service market. Against each pattern we study the corresponding SLA design scheme that can meet its unique requirements. In addition, we systematically investigate the application of Bayes estimator in these schemes, since the knowledge of their negotiation counterpart or market competitors is essential for reaching the goal of utility optimization. At the end, we cast the hybrid SLA design framework into a stochastic model that allows decision makers to obtain evaluations of performance of interest.

CHAPTER 1. Introduction

Services computing has two main objectives: maximize the utilization of the existing business assets and increase the capability of satisfying the ever-changing market requirements, which are essential to ensure the sustainable competitive edge in the global supply chain. An agile business product line should be able to horizontally span company walls while supporting multiple business functionalities and multiple types of users. The ever-increasing computational power and the wide deployment of internet service provide strong support to help the enterprises achieve the goal of services computing.

Service-oriented architecture (SOA) provides a flexible and extensible platform to instantiate the business management philosophy of services computing. It introduces two primary business advantages [1][2]:

- The adoption of SOA-based resource management architecture enables the business firm to save cost by leveraging existing software and hardware assets.
- The embedded universally agreed-upon specifications allow the enterprises to deliver various types of business applications through the enterprise-wide service composition or business-to-business application integration in an automatic way.

Thriving and diverse service-oriented applications across different business spectrums motivate us to address the related theoretical research from a deeper business oriented thought. Pertaining to this vision, we formalize the concept of Competitive Service Market (CSM) as an generic framework to study the business dynamics for various service systems implemented either on the Web or on other platforms [3].

1.1 Background

A competitive service market is formulated as $CSM = \{\mathcal{ASP}, \mathcal{CSP}, \mathcal{AS}\}$, where

- $\mathcal{ASP} = \{asp_i : i = 1, \ldots, \|\mathcal{ASP}\|\}$: $\mathcal{ASP}$ is a set of atomic service providers, denoted by asp_i, which is assumed to embed with self-contained functionality without needing the cooperation of other agents to provide a specific type of service.
- $\mathcal{CSP} = \{csp_i : i = 1, \ldots, \|\mathcal{CSP}\|\}$: $\mathcal{CSP}$ is a set of composite service providers, denoted by csp_i, which can invoke several atomic service providers in a service composition process.
- $\mathcal{AS} = \{AS_i : i = 1, \ldots, \|\mathcal{AS}\|\}$: $\mathcal{AS}$ is a set of atomic services existing in the market.

Here, $\|\mathcal{ASP}\|, \|\mathcal{CSP}\|$ and $\|\mathcal{AS}\|$ stand for the sizes of sets $\mathcal{ASP}, \mathcal{CSP}$ and $\mathcal{AS}$, respectively. Atomic services act as the basic units in a service system. The realization of their functionalities does not require the involvement of other units. On the other hand, service composition protocol aggregates several atomic services into a composite service controlled by a given business logic and then delivers it to the customer [4]. The service composition process enables a collection of heterogeneous service providers to collaborate in a virtual coalition that can span across different enterprises. In the CSM setting, the service composition protocol is owned and deployed by the composite service providers, which aslo act as the interface between the consumers and the market, whereas atomic service providers are hidden from the consumers.

Figure.1.1 illustrates an example of a competitive service market. The market has two composite service providers: *composite service provider 1* and *composite service provider 2*. Through the service composition protocol, *composite service provider 1* can invoke the atomic services provided by *content service 1*, *information system 2*, *information system 3* and *marketing service* while *composite service provider 2* can invoke the atomic

services provided by *content service 2*, *information system 3*, *information system 1* and *marketing service*.

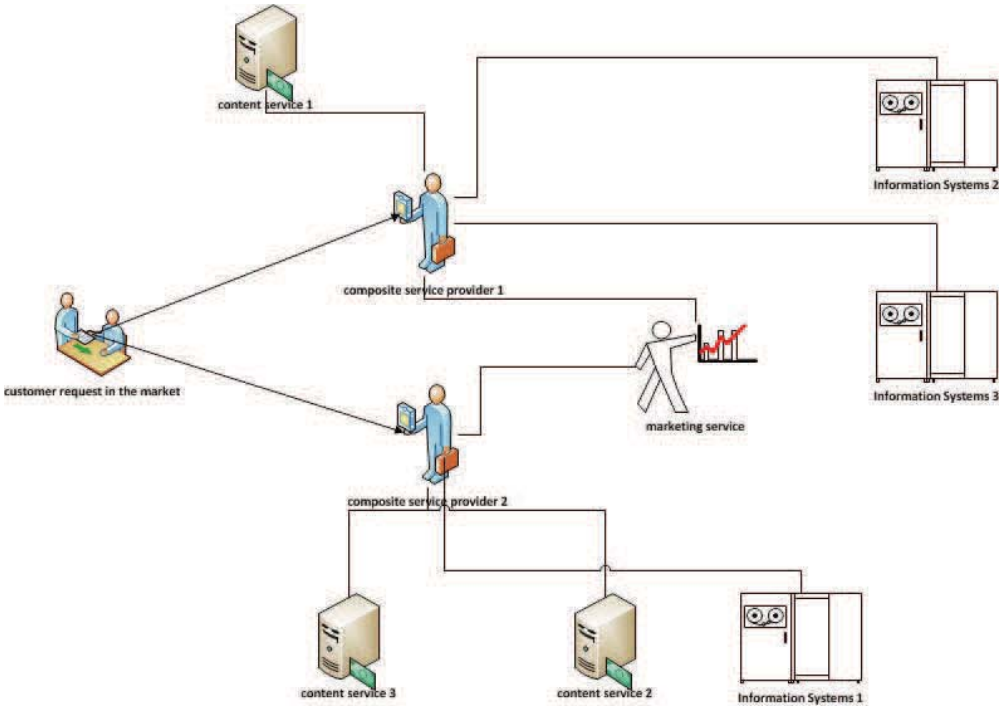

Figure 1.1 An example of competitive service market

Figure.1.2 displays a composite service that is able to be composed by *composite service provider 2*. This composite service describes the workflow of a content service provisioning process. In Figure.1.2, the *information system 2* is an information infrastructure service provider which provides storage space for electronic medias, such as movies and on-line games. The *content service 1* is a content service provider which can add unique features developed in-house to the media based on customers' various requirements. A specialized marketing consultant, i.e., *marketing service* shown in Figure.1.2 is also hired to improve the market share and manage the customer service.

From the topological point of view, each composite service can be treated as a workflow graph, in which a vertex represents an atomic service and an edge stands for the busi-

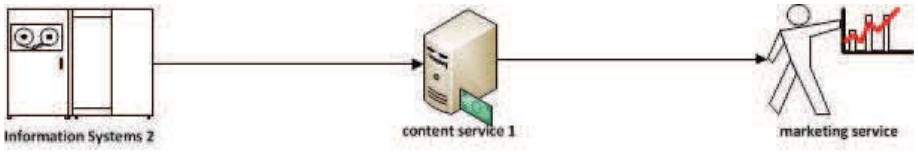

Figure 1.2 An example of composite service

ness logic underlying the composite service. Figure. 1.3 illustrates a composite service example consisting of eight atomic services, which is denoted as $\mathscr{S} = \{AS_1, AS_2, \ldots, AS_8\}$. In our research, we use $\mathscr{S}$ to denote the composite service.

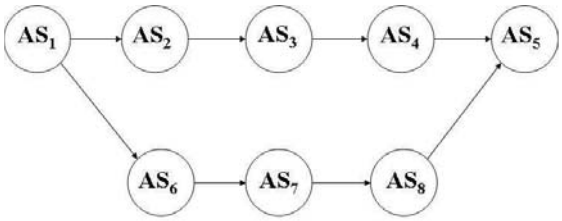

Figure 1.3 An example of composite service

With respect to a given composite service provider, csp_o, we define the set of atomic service providers maintaining the business relationship with csp_o as *controlled group* denoted as $\mathcal{CG}_o$. The business relationship between a composite service provider and an atomic service provider belonging to its controlled group is framed as the service level agreement (SLA) which will be investigated in detail later. Figure.1.4 shows a conceptual diagram of csp_o and its controlled group. In our research, we use $\mathcal{CG}_o^t$ to represent a set of atomic service providers which belong to $\mathcal{CG}_o$ and provide the same type of atomic service identified with superscript t.

In the example shown in Figure.1.4, we have $\mathcal{CG}_o = \mathcal{CG}_o^1 \cup \mathcal{CG}_o^2 \cup \mathcal{CG}_o^3 \cup \mathcal{CG}_o^4$. In set $\mathcal{CG}_o^1$, there are three atomic service providers: asp_1,asp_2 and asp_3 which can provide atomic service, AS_1. From the viewpoint of a composite service provider, the existence of multiple candidates makes it necessary to develop a strategy that decides which atomic service provider should be selected for a given type of atomic service. This is the main objective

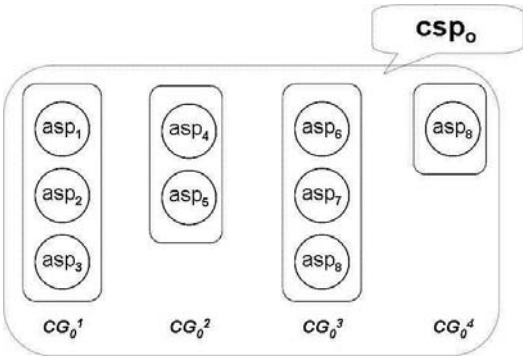

Figure 1.4 Composite service provider and its controlled group

of the service composition protocol and has been extensively studied from various perspectives [5][6]. From the point of view of an atomic service provider, the existence of other agents sharing the same functionality constitutes a potential competition, which asks more strategic decision capabilities if it intends to generate profit from the service provisioning process.

1.2 Management Issues for Service Market

Figure.1.5 displays the conceptual diagram of the management architecture for a service market. A set of functional components are connected through the information flow channel.

- A large amount of atomic service providers register their profiles in the market. These profiles are stored in the *service registry* and provide service attributes as the inputs for the service composition protocol. Once the number of stored service profiles increases, the related indexing and discovery tasks can become quite challenging. In [7], we develop a hierarchical service structuring approach based on pattern recognition techniques. In [8], we design the counting Bloom filter-based service storage architecture and develop multi-level service discovery protocol based on this architecture.

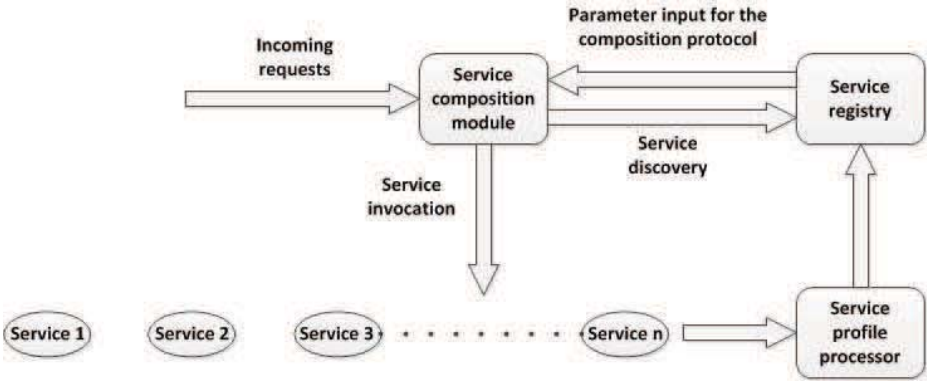

Figure 1.5 Management architecture of service market

- The module of *Service profile processor* extracts various types of information from the input service profile, and perform necessary operations before storing a service profile in the registry. For instance, in order to use the numerical-based pattern recognition algorithms, the service profiles have to be quantized before a clustering algorithm can be applied.

- The *Service composition module* is embedded with a set of optimization algorithms to discover and aggregate several services into the target business process. The service profile parameters stored in the service registry are used as the inputs for the optimization algorithms. In the service market, the service composition process affects the business interests of multiple service agents at different levels. For instance, the composite service provider focuses on the end-to-end performance of the whole business process while an atomic service provider tends to maximize its own interest. One of the challenges of these optimization algorithms is to balance the business interests of different service agents in accordance with a set of pre-determined objectives [5].

1.3 Structure the Competitive Service Market From the Perspective of Supply-and-Demand Relationship

In the competitive service market, atomic service providers are the upper stream services suppliers who manage the underlying service infrastructures and deliver the services to the composite services providers in accordance with the pre-designed SLAs. On the other hand, a composite services provider plays a twofold role:

- From the perspective of an outside service consumer, the composite service provider behaves like a seller of composite service.
- From the viewpoint of atomic service provider selling services, the composite service provider behaves like a purchaser of atomic service.

In the CSM setting, the above supply-and-demand relationships are delineated by the following constraints in terms of the requirements analysis of the requested composite service, e.g., $\mathscr{S} = \{AS_1, \ldots, AS_m\}$.

Functionality constraint : From the perspective of functional requirements, the functionalities associated with every participating atomic service should have at least one candidate service provider, which indicates that $\|\mathcal{CG}_o^i\| > 0 \quad \forall\ AS_i \in \mathscr{S}$, Where $\|\cdot\|$ represents the number of candidates.

Quality of service(QoS)-based constraint : In reality, every incoming service request is subject to certain QoS requirements along with several dimensions, such as availability and the response time. These various QoS requirements are framed as either global QoS constraints or local QoS constraints in terms of their respective objectives. For instance, csp_o is scheduled to deliver $\mathscr{S}$ illustrated in Figure. 1.3. A local QoS constraint is that the availability for the participating atomic service AS_1 needs to be at least 90%; and a global constraint is that the end-to-end response time should be less than 8 hours.

Profit-based constraint : In order to deliver the required service, the composite service provider has to purchase atomic services from related service providers, which naturally incurs costs. The budget of a service consumer, γ represents the upper bound for the total cost of the composite service provider if it wants to generate some non-negative profit from this service delivery process. This profit-based constraint is formulated as

$$\sum_{j=1}^{j=m} C_{o,i}^{j} \leq \gamma \quad \exists\ asp_i \in \mathcal{CG}_o^j \ \ \forall AS_j \in \mathscr{S}. \tag{1.1}$$

$C_{o,i}^{j}$ represents the transaction cost between composite service provider csp_o and an atomic service providers belonging to $\mathcal{CG}_o^j$ which is able to deliver the functionality associated with AS_j. If Equation.1.1 is not satisfied, then there does not exist any economically feasible solution for the composite service provider to deliver the service request $\mathscr{S}$.

Among the three major constraints usually involved in the service composite process, both QoS-based constraint and the profit-based constraint are within the scope of non-functional requirements. Mathematically, a service composition process can always be framed as an optimization problem that has to satisfy these three constraints.

1.4 Research Challenges and Contributions

Managing the competitive service market not only involves a set of decision making processes with different objectives and instantiations, but also requires the construction of the underlying service infrastructure to support the diverse business activities in the market. Our research focuses on three fundamental issues of the service management, which are listed as follows:

1. The decision making processes rely on quantitative models to help predict the performance of various types of market players. In the competitive service market,

the market players differ in the position they hold in the market and their respective system features. Moreover, these market players interact with each other through various types of business relationships. How to capture these differences and their interrelations in the modeling effort is the first issue we will address.

2. Each service provider can support multiple quality configurations for a single type of service by controlling the performance of its service facility. This feature is essential to gaining a competitive edge in the service market. Therefore, a service provider tends to register every possible quality configurations in the service registry. As a consequence, a scalable storage space is required to store these service parameter configurations and support the increased complexity of service discovery process.

3. The business relationships between different market players in the competitive service market are framed through the service level agreement (SLA). A signed SLA stipulates the service cost and service qualities, which are essential for searching, delivering and monitoring the realized services. In the competitive service market, the ever-changing business scenarios can affect the framing process of the business relationship, i.e., SLA design protocol. How to design SLAs in different business scenarios and understand the dynamic business relationships framed by different SLA design protocols is the third issue we will address.

The contributions of this research are summarized as follows:

1. We propose the concept of a competitive service market in order to capture the business dynamics underlying the service systems. We characterize the service chain in the market based on the formalized structural hierarchy and the constrained supply-and-demand relationship.

2. We systematically discuss the modeling issues of the service market.

- In accordance with the hierarchical structure of the service market, we investigate the modeling approaches at both atomic service level and composite service level, along with market level.
- We discuss the influences of heterogeneous time scales between the composite service and atomic service on the calculation of end-to-end response time. We develop an algorithm to solve this issue by capturing the time-dependent behavior.

3. In accordance with the service requirements analysis, we classify the concept of service and the concept of service parameter configuration in terms of the functional and non-functional features.
   - In the context of the classification of service and its service parameter configurations, we divide the service discovery process into a two-step procedure.
   - Based on the chaining relationship connecting service, service parameter configuration and service provider, we develop a scalable storage architecture consisting of a centralized service array and a set of decentralized storage units using the entries in the service array as index.
   - We apply counting Bloom filter to design storage unit and develop the related service discovery process using the structural properties of counting Bloom filter.
   - We come up an analytically solvable metrics to predict the performance of the service discovery process on this system.
4. The business relationship between different market players are framed by SLAs.
   - We identify several SLA design patterns in accordance with different business scenarios that can occur in the life cycle of the market.

- We develop SLA design methodology with respect to each identified SLA design pattern.
- We investigate the application of Bayes estimator in the bilateral negotiation-based multi-attributes SLA design.
- We investigate the target-oriented SLA design in the sequential auction using Bayes estimator.
- We build a Markov chain model to study the dynamic business relationships framed by coexisting SLA design protocols. The constructed model will help different market players make both strategic and operational decisions.

1.5 Dissertation Organization

The rest of this dissertation is organized as follows. In Chapter 2, we discuss the modeling approaches in accordance with the hierarchical structure of the competitive service market. We not only address the modeling issues at the atomic service level but also investigate how to model the market as a whole against a specific performance metrics. In Chapter 3, we propose a scalable storage architecture design, and investigate the associated service discovery protocol. In Chapter 4, we identify different SLA design patterns in terms of their respective business scenarios, and develop the corresponding design methodologies. We conclude this dissertation with summary and future work in Chapter 5.

CHAPTER 2. A Systematic View of Service Market

2.1 Introduction

Within the competitive service market, each market player needs to make various types of strategic and operational decisions. Strategic decisions tend to cover the long and intermediate time horizon, such as the service infrastructure investment and the placements of service resources. On the other hand, the operational decisions focus on each single operation instance, such as the service selection and composition. Making these decisions requires the modeling efforts at both the localized level,i.e., atomic service provider itself, and the global level, i.e., composite service provider spanning across the enterprise boundaries. The constructed model and related performance analysis guide the allocation, pricing and management of various types of service resources. The first part of this chapter presents a queuing modeling framework for the atomic service provider. The introduction of queuing theory enables the quantitative analysis of a variety of service parameters, which are usually used as the inputs to other modeling problems. The second part of this chapter is going to discuss a modeling formulation at the global level of the competitive service market. The contents of this chapter are the extension of our previous work published in [9].

2.2 Queuing Model for Atomic Service Provider

Atomic service provider performs as the building block for the service market, and the stochastic modeling of which forms the basis for understanding the competitive service market as a whole.

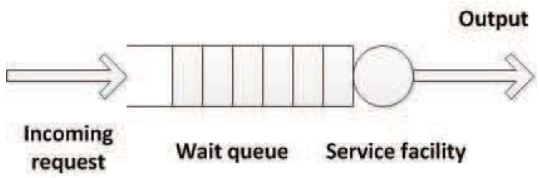

Figure 2.1 Queuing model of atomic service provider

Figure.2.1 illustrates a queuing model for an atomic service provider. The jobs will accumulate in the wait queue once the arriving rate of incoming requests overwhelms the service capacity. Interarrival times, service rate and buffer capacity are three essential metrics to characterize a queue. In this research, the following assumptions are made.

- The interarrival times follows an exponential distribution.
- The service processing time does not fit into any specific probabilistic models. The statistical analysis against the historical record can provide the cumulative distribution function.
- The system can have no more than K service requests at one time. In other words, the capacity of the wait queue is $K - 1$.

A large portion of queuing-based studies tends to assume the Poisson distribution for both service arriving and processing processes. Due to the heterogeneous service infrastructure, which can be hardware, software or even the human being, we relax the service processing model to the general probability distributions. In terms of the queuing modeling framework, this type of stochastic system can be modeled as a M/G/1/K

queue. From the perspective of theoretical study on the queuing modeling formalism, more complicated queuing models can be created to allow higher degree of relaxations. However, a large portion of these queuing models cannot be solved analytically and only the numerical simulations are available for the investigation. In the following, we discuss how some essential performance metrics can be derived based on the queuing model.

Response time is an essential metric that is required by the composite service provider to plan the service composition process and estimate the system performance beforehand. Moreover, response time also functions as a basic element for deriving other performance metrics, e.g., reliability. The expected response time for the atomic service provider asp_i to deliver service AS_j is denoted as $\mathcal{T}_j^i$. The event density of asp_i is characterized by $\mathbf{f}_{\eta_i}(t)$, a probability density function to describe a random variable representing the functioning time before the failure [10]. Hence, we have

$$\mathscr{R}_{i,j} = 1 - \int_0^{\mathcal{T}_j^i} \mathbf{f}_{\eta_i}(t)\ dt, \tag{2.1}$$

where $\mathscr{R}_{i,j}$ represents the reliability for the service delivery process of asp_i over service AS_j.

The system dynamics of the M/G/1/K queue is characterized by the number of jobs staying in the system. Let $\mathbf{P}_i : i = 1, ..., K$ denote the probability with which an admitted request sees i requests in the wait queue and service facility as a whole. $\mathbf{P}_K$ thus stands for the blocking probability. According to the queuing theory,

$$\mathbf{P}_i = \frac{\mu\pi_i}{\lambda + \mu\pi_0} \quad i = 0, ..., K-1 \quad \text{and} \quad \mathbf{P}_K = 1 - \frac{\mathbf{P}_o}{\pi_0} \tag{2.2}$$

In Equation.2.2, π_i is the stationary probability distribution of the embedded Markov chain underlying the M/G/1/K queuing system. Please refer to [11][12] for more details. Here, we just want to stress the fact that $\mathbf{P}_i$ is analytically solvable and depends only on the system parameters: λ, μ and K. Based on Eq.2.2, the expected number of requests

residing in the queue is calculated as

$$M = \sum_{i=0}^{K} i\mathbf{P}_i. \tag{2.3}$$

According to the Little's law [12], i.e., the average number of customers in the system equals to the product of the average response time and the average effective arrival rate. Hence, the mean service response time for an atomic service provider can be computed by

$$\mathcal{T} = \frac{M}{\lambda(1 - \mathbf{P}_K)} \tag{2.4}$$

In Equation.2.4, $(1 - \mathbf{P}_K)$ is the admission probability and hence $\lambda(1 - \mathbf{P}_K)$ represents the effective arrival rate. Submitting the computed $\mathcal{T}$ into Equation.2.1 leads to the queuing-based reliability metric. The analysis presented in this section is to provide a building block for the further system analysis of the service market. Based on the queuing framework, we can compute a variety of performance metrics which are used as the input parameters for the decision making processes of different market players.

2.3 Recursive Procedure for Computing End-to-End Response Time in the Competitive Services Market

Instead of focusing on a single atomic service provider, in this section, we consider the performance analysis at the composite service level, which is the calculation of the end-to-end response time for a composite service. We generalize the studied model from the trivial scenario of a single service provider to the scenario involving the existence of multiple candidate service providers, and finally we take into account the time-dependent behavior of the involved atomic service providers.

Algorithm.2.1 lists a recursive procedure for computing the end-to-end response time for the composite service, like the one shown in Figure.2.2. $\mathcal{T}^i$ denotes the response time for service AS_i. Since this algorithm does not consider the existence of multiple

Algorithm 2.1 $\mathbf{FTA}_1(AS_i)$:Recursive procedure for calculating the response time for a composite service

1: **if** $\mathcal{N}(AS_i) = \emptyset$ **then**
2: $\quad \mathcal{T}^i = \mathbf{Q}(asp_j)$;
3: **else**
4: $\quad \mathcal{T}^i = \mathbf{Q}(asp_j) + \text{MAX}(\mathbf{FTA}_1(AS_j) : \forall AS_j \in \mathcal{N}(AS_i))$;
5: **end if**
6: **return** $\mathcal{T}^i$;

candidates for a single service, we just assume that AS_i is delivered by asp_i and use $\mathbf{Q}(asp_i)$ to denote the expected service time computed from the queuing model of asp_i. $\mathcal{N}(AS_i)$ represents the atomic service next to AS_i in the business logic. For instance, $AS_2 \in \mathcal{N}(AS_1)$ and $AS_3 \in \mathcal{N}(AS_1)$ in Figure.2.2.

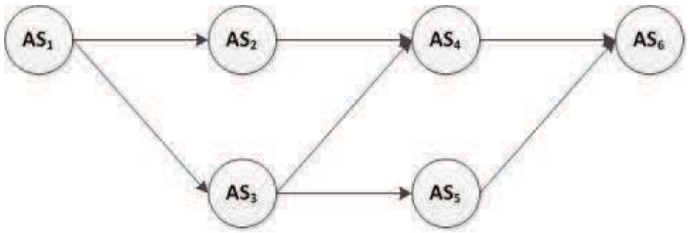

Figure 2.2 An example for illustrating the computation procedure

The procedure is launched by calling the initialization service of the studied composite service. To compute the response time for the composite service shown in Figure.2.2, we call $\mathbf{FTA}_1(AS_1)$. In accordance with the business logic shown in Figure.2.2,

$$\mathbf{FTA}_1(AS_1) = \mathcal{T}^1 = \mathbf{Q}(asp_1) + \text{MAX}(\mathbf{FTA}_1(AS_2), \mathbf{FTA}_1(AS_3)) \tag{2.5}$$

And

$$\begin{aligned} \mathbf{FTA}_1(AS_2) &= \mathbf{Q}(asp_2) + \mathbf{FTA}_1(AS_4) \\ \mathbf{FTA}_1(AS_3) &= \mathbf{Q}(asp_3) + \text{MAX}(\mathbf{FTA}_1(AS_4), \mathbf{FTA}_1(AS_5)) \end{aligned} \tag{2.6}$$

Since no service node is next to the AS_6,

$$\begin{aligned} \mathbf{FTA}_1(AS_4) &= \mathbf{Q}(asp_4) + \mathbf{Q}(asp_6) \\ \mathbf{FTA}_1(AS_5) &= \mathbf{Q}(asp_5) + \mathbf{Q}(asp_6) \end{aligned} \tag{2.7}$$

The results given by Equation.2.7 will be returned back to Equation.2.6 and finally to Equation.2.5 to get $\mathbf{FTA}_1(AS_1) = \mathcal{T}^1$ according to the invoking sequence in this recursive procedure.

The setting of competitive service market allows the existence of multiple atomic service provider candidates for a single service. The procedure of $\mathbf{FTA}_1$ does not consider this kind of possibility. Therefore, we design a refined procedure identified as $\mathbf{FTA}_2$.

Algorithm 2.2 $\mathbf{FTA}_2(AS_i)$:Recursive procedure for calculating the composite service response time in the services market

1: **if** $\mathcal{N}(AS_i) = \emptyset$ **then**
2: $\quad \mathcal{T}^i = \text{MIN}(\mathbf{Q}(asp_j) : \forall asp_j \in \mathcal{CG}^i_o)$;
3: **else**
4: $\quad \mathcal{T}^i = \text{MIN}(\mathbf{Q}(asp_j) : \forall asp_j \in \mathcal{CG}^i_o) + \text{MAX}(\mathbf{FTA}_2(AS_j) : \forall AS_j \in \mathcal{N}(AS_i))$;
5: **end if**
6: **return** $\mathcal{T}^i$;

Algorithm.2.2 differs with Algorithm.2.1 at lines 2 and 4, where the former has to take into account the existence of multiple candidates. Both $\mathbf{FTA}_1(AS_i)$ and $\mathbf{FTA}_2(AS_i)$ assume that performance metrics of the atomic service provider asp_i does not vary with time. In practice, the life cycle of a composite service and the one of an atomic service may have different time scales. Consider the example shown in Figure.2.3, AS_1 and AS_2 can take multiple days, and AS_4 only needs multiple minutes for fulfilling its functionality. This is a system consisting of components which differ in their time scales. If the atomic service providers available for delivering AS_4 are un-stationary systems whose performance metrics are generally time-dependent functions, the difference between the starting time of the composite service and the starting time of AS_4 should be tracked

in order to compute the end-to-end response time by using the accurate response time of AS_4, which depends on when AS_4 will be invoked.

Figure 2.3 An example illustrating the time difference

A refined end-to-end response time computation procedure is listed in Algorithm.2.3.

Algorithm 2.3 $\mathbf{FTA}_3(AS_i, t)$:Recursive procedure for calculating the composite service response time considering queuing factor

1: **if** $\mathcal{N}(AS_i) = \emptyset$ **then**
2: $\quad \mathcal{T}^i = \text{MIN}(\mathbf{QT}(asp_j, t) : \forall asp_j \in \mathcal{CG}_o^i)$;
3: **else**
4: $\quad t_1 = \text{MIN}(\mathbf{QT}(asp_j, t) : \forall asp_j \in \mathcal{CG}_o^i)$;
5: $\quad \mathcal{T}^i = t + t_1 + \text{MAX}(\mathbf{FTA}_3(AS_j, t + t_1) : \forall AS_j \in \mathcal{N}(AS_i))$;
6: **end if**
7: **return** $\mathcal{T}^i$;

Following similar procedures, the end-to-end response time for a given composite service can be calculated by calling $\mathbf{FTA}_3(AS_1, 0)$, where AS_1 is the initialization service node and thus the elapsed time to AS_1 is 0. $\mathbf{FTA}_3(AS_i, t)$ differs with $\mathbf{FTA}_1(AS_i)$ and $\mathbf{FTA}_2(AS_i)$ in terms of the following two points:

- The service time computation procedure, invoked at line 2, is identified as **QT**, which takes two parameters, asp_i and t. t is the elapsed time inherited from the recursive call at line 5. Based on this information, each asp_i can track the elapsed time between the starting time of the composite service and the starting time of this specific atomic service to be delivered by itself in accordance with the business logic. Therefore, more accurate service information, such as the blocking probability or service time, can be supplied to the composite service provider to help the service selection process. On the other hand, $\mathbf{FTA}_1(AS_i)$ and $\mathbf{FTA}_2(AS_i)$ do not consider this type of time information.

- The recursive call at line 5 monitors the elapsed time from the initialization point to its finishing point. The resulting sum, $t + t_1$ is used as the parameter of the recursive call for the next service node in the business logic.

Starting from a trivial procedure of computing the end-to-end response time for composite service, we step-by-step refine this procedure by taking into account two important factors in the practical competitive service market modeling: multiple candidates for a single atomic service and the time-dependent behavior of the atomic service providers.

2.4 Market-based Availability Modeling

Section 2.2 focuses on the system performance modeling at the atomic service level, and section 2.3 considers an important performance metrics at the composite service level. In this section, we are going to discuss the stochastic analysis of a performance metrics at the market level using Markov chain which is a random process well suited to model the stochastic systems.

2.4.1 Discrete-Time Markov Chains

Markov chain is a particular type of random process, and is characterized by the memoryless property, i.e., the state in the future only depends on the current state [13]. The Markov chain is represented as $\{X(t), t \in T\}$, where $X(t)$ is a numerical value, and its domain is referred to as the *state space*. In the Markov chain theory, the state space is always discrete. T is the parameter space and it can be either discrete or continuous. In the discrete parameter space, t is usually called as *step*. In the continuous parameter space, t is usually called as *time*. A Markov chain with discrete parameter space is referred to as discrete-time Markov chain (DTMC). The DTMC dynamics is controlled by the one-step transition probability $\mathbf{P}_t[i, j]$. It stands for the probability of reaching

state j from i in one step and is given in Equation.2.8.

$$\mathbf{P}_t[i,j] = \mathbf{Pr}\{X(t+1) = j | X(t) = i\} \tag{2.8}$$

The value of $\mathbf{P}_t[i,j]$ is controlled by three factors: t, i and j. The homogeneous DTMC removes the effect of t, which is

$$\mathbf{P}_{t_1} = \mathbf{P}_{t_2} \quad \forall \quad t_1 \in T \quad \text{and} \quad t_2 \in T \tag{2.9}$$

For the homogeneous Markov chain, we can just remove t from the transition probability for simplicity. Figure.2.4 shows a DTMC example composed of nine states, $\{A, B, C, D, E, F, G, H, I\}$. The value associated with each transition represents the one-step transition probability.

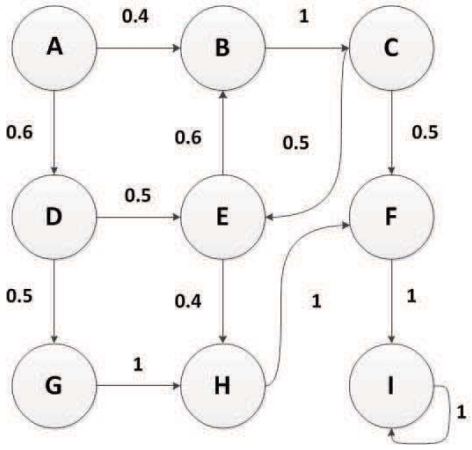

Figure 2.4 A discrete-time Markov chain

With respect to the DTMC shown in Figure.2.4, we number the states $\{A, B, C, D, E, F, G, H, I\}$ as $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ respectively. The corresponding matrix representa-

tion is given in the following.

$$\mathbf{P} = \begin{bmatrix} 0 & 0.4 & 0 & 0.6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 & 0.5 & 0 & 0 \\ 0 & 0.6 & 0 & 0 & 0 & 0 & 0 & 0.4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

.

The Markov chain dynamics at each step can be captured by a probability vector:

$$P_t = [P_t[1], ..., P_t[m]] \quad \text{and} \quad P_t[i] = \mathbf{Pr}\{X(t) = i\} \tag{2.10}$$

In the above equation, m represents the number of states in the Markov chain model. The initial probability vector P_0 at step 0 describes the system dynamics at the very beginning, which is usually given as an assumption by the system modeler. The combination of initial probability vector and the probability transition matrix uniquely determines the probability vector P_t for every t [14]. This relationship is implemented through a series of matrix multiplication.

$$P_t = P_{t-1} \times \mathbf{P} \Longrightarrow P_t = (P_{t-2} \times \mathbf{P}) \times \mathbf{P} \Longrightarrow P_t = P_0 \times \mathbf{P}^t \tag{2.11}$$

2.4.2 Market-Based Availability Modeling Using Markov Chains

Market-based availability is an important performance metrics for evaluating the service market as a whole. With respect to a request,$\mathscr{S} = \{AS_1, \ldots, AS_i, \ldots, AS_n\}$, its market-based availability is measured as the probability for the request service to be able to be fulfilled by at least one composite service provider in the market.

Figure 2.5 A requested composite service with sequential pattern

Figure 2.5 shows a composite service example of $\mathscr{S} = \{AS_1, AS_2, AS_3\}$, which has a sequential workflow pattern. With respect to this service request, we are going to compute the availability for a market consisting of two composite service providers: csp_1 and csp_2.

We model the system dynamics by defining the state as $A =$ "$a_1a_2a_3$". a_1 represents the atomic service that is being processed and it can have value 1, 2 or 3. a_2 represents whether csp_1 is able to deliver service S_{a_1} from its controlled atomic service providers. If csp_1 is able to fulfill the task, than $a_2 = 1$; otherwise, $a_2 = 0$. Similarly, a_3 represent whether csp_2 is able to deliver service AS_{a_1} from its controlled atomic service providers. For instance, state 100 represents that both csp_1 and csp_2 fail to fulfill AS_1 from their respective controlled atomic service providers; state 311 indicates that both csp_1 and csp_2 can fulfill AS_3.

Figure.2.6 displays the constructed Markov chain model consisting of 12 states:$\{100, 101, 110, 111, 200, 201, 210, 211, 300, 301, 310, 311\}$. The constructed Markov model has a layered topological structure which has following features:

- State $i01$ can only reach the states $(i+1)01$ and $(i+1)00$;
- State $i10$ can only reach the states $(i+1)10$ and $(i+1)00$;
- State $i11$ can reach all of the states in the next layer which is composed of the states represented as $(i+1)a_2a_3$.

Forming this type of layered topological structure comes from the following facts:

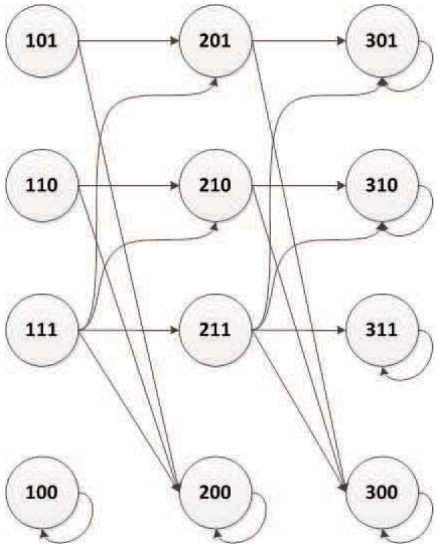

Figure 2.6 The Markov chain model of the market-level availability for service request $\mathscr{S} = \{AS_1, AS_2, AS_3\}$

- If a composite service provider fails to fulfill the i-th atomic service, then it is meaningless for it to fulfill the (i+1)-th atomic service.
- If a composite service provider successfully finishes the i-th atomic service, it can either successfully fulfill or fail to deliver the (i+1)-th atomic service.

After fixing the topological structure of a Markov chain model, we still need to obtain its one-step transition probabilities which are obtained by analyzing the probabilistic behaviors of the studied system.

The fact that state $(i+1)01$ is reachable from state $i01$ indicates that csp_2 can deliver the service AS_{i+1}. Let $\mathbf{Pr}^f(\mathcal{CG}_2^{i+1})$ denotes the probability that none of the atomic service providers belonging to $\mathcal{CG}_2^{i+1}$ is available to serve AS_{i+1}. In chapter 1, $\mathcal{CG}_2^{i+1}$ stands for the set of atomic service providers controlled by csp_2 for providing AS_{i+1}. $\mathbf{Pr}^f(\mathcal{CG}_2^{i+1})$ can be calculated as follows.

$$\mathbf{Pr}^f(\mathcal{CG}_i^j) = \prod \mathbf{Pr}^f(asp_l) \quad \forall asp_l \in \mathcal{CG}_i^j \tag{2.12}$$

In Equation.2.12, $\mathbf{Pr}^f(asp_l)$ denotes the probability of failure for asp_l. Hence, we have

$$\mathbf{P}[i01,(i+1)01)] = 1 - \mathbf{Pr}^f(\mathcal{CG}_2^{i+1}). \tag{2.13}$$

From Equation.2.13, the transition probability of $\mathbf{P}[i01,(i+1)00)]$ is computed as

$$\mathbf{P}[i01,(i+1)00)] = 1 - \mathbf{P}[i01,(i+1)01)] = \mathbf{Pr}^f(\mathcal{CG}_2^{i+1}) \tag{2.14}$$

Similarly, we have

$$\mathbf{P}[i10,(i+1)10)] = 1 - \mathbf{Pr}^f(\mathcal{CG}_1^{i+1}) \tag{2.15}$$

$$\mathbf{P}[i10,(i+1)00)] = \mathbf{Pr}^f(\mathcal{CG}_1^{i+1}) \tag{2.16}$$

The fact that state $(i+1)11$ is reachable indicates that at least one of the atomic service providers in $\mathcal{CG}_1^{i+1}$ and at least one of the atomic service providers in $\mathcal{CG}_2^{i+1}$ are able to serve AS_{i+1}. Therefore,

$$\mathbf{P}(i11,(i+1)11) = (1 - \mathbf{Pr}^f(\mathcal{CG}_1^{i+1}))(1 - \mathbf{Pr}^f(\mathcal{CG}_2^{i+1})) \tag{2.17}$$

The fact that $(i+1)00$ is reachable indicates that none of the atomic service providers in $\mathcal{CG}_1^{i+1}$ and none of the atomic service providers in $\mathcal{CG}_2^{i+1}$ is able to serve AS_{i+1}. Therefore,

$$\mathbf{P}(i11,(i+1)00) = \mathbf{Pr}^f(\mathcal{CG}_1^{i+1})\mathbf{Pr}^f(\mathcal{CG}_2^{i+1}) \tag{2.18}$$

The similar analysis leads to the other remaining transition probabilities:

$$\mathbf{P}(i11,(i+1)01) = \mathbf{Pr}^f(\mathcal{CG}_1^{i+1})(1 - \mathbf{Pr}^f(\mathcal{CG}_2^{i+1})) \tag{2.19}$$

$$\mathbf{P}(i11,(i+1)10) = (1 - \mathbf{Pr}^f(\mathcal{CG}_1^{i+1}))\mathbf{Pr}^f(\mathcal{CG}_2^{i+1}) \tag{2.20}$$

Among the twelve states shown in Figure.2.6, states $\{100, 200, 300, 301, 310, 311\}$ are absorbing states. Once reaching an absorbing state, the system will not leave. With respect to these six absorbing states, the states $\{100, 200, 300\}$ represent the scenarios where the market fail to meet the request of $\mathscr{S} = \{AS_1, AS_2, AS_3\}$; and the states

$\{301, 310, 311\}$ represent the scenarios where the market is able to meet the request of $\mathscr{S} = \{AS_1, AS_2, AS_3\}$. The Markov chain model shown in Figure.2.6 belongs to a special type of Markov chain, which is called as absorbing Markov chain [13]. We use $\pi[i, j]$ to denote the probability of being absorbed in an absorbing state j once starting from the transient state j.

$$\phi = \pi_a(\text{“301”}) + \pi_a(\text{“310”}) + \pi_a(\text{“311”}) \tag{2.21}$$

In Equation.2.21, ϕ represents the market-based availability; $\pi_a(\text{“301”}), \pi_a(\text{“310”})$ and $\pi_a(\text{“311”})$ represent the aggregation absorption probability for states “301”, “310” and “311” respectively. Their computations are given as follows:

$$\begin{aligned} \pi_a(\text{“301”}) \;=\; & \pi(\text{“101”}, \text{“301”})P_0[\text{“101”}] + \pi(\text{“110”}, \text{“301”})P_0[\text{“110”}] \\ & + \pi(\text{“111”}, \text{“301”})P_0[\text{“111”}] + \pi(\text{“100”}, \text{“301”})P_0[\text{“100”}] \end{aligned} \tag{2.22}$$

In Equation.2.22, $\pi(\text{“101”}, \text{“301”})$ represents the absorption probability of state “301” given a start state of “101”. $P_0[\text{“101”}]$ represents the initial probability of state “101”, which is an input parameter given by the system modeler. For the Markov chain model shown in Figure.2.6, only states “101”, “110” “111” and “101” can have nonzero values in the initial probability vector. This is why we only consider four starting states in Equation.2.22. The computation of $\pi(\text{“101”}, \text{“301”})$ can use the standard procedures based on fundamental matrix [15]. The physical meanings for other terms in Equation.2.22 and the computation of $\pi_a(\text{“310”})$ and $\pi_a(\text{“311”})$ are similar.

2.5 Summary and Discussion

Competitive service market is proposed to study the interactions and competitive natures among customers, atomic service providers and composite service providers. The

research reported in this chapter aims to understand this relationship from the system analysis perspective. We structure our modeling efforts at two levels in accordance with the inherent hierarchy in the service market, which has the atomic service at the bottom level and the composite service at the upper level. We investigate different modeling approaches for each level. The queuing modeling at the atomic service level functions as the building block for other analyses, while the market-based availability analysis using DTMC provides an illustrative yet operational example for the stochastic analysis at the market level. Performing this availability analysis aims to illustrate how a generic service system is modeled and analyzed. Detailed probabilistic analysis has been made to construct the transition matrix for the Markov chain model. Based on the constructed Markov chain, we investigate how a performance metric, like availability, can be analyzed over the service market. In practice, the concrete Markov model varies with the studied system and the requested service. We can project that the size of the state space can become quite large when the composite service involves a lot of atomic services. The next step research along with this direction is to investigate the applications of high-level modeling formalisms on the performance analysis of service market [16]. The end-to-end response time computation procedure demonstrates how the inherent complexities in the composite service can be taken into account. Modeling the time-dependent behavior for a service system represents another challenge that has to be meet in our future research [17].

CHAPTER 3. A Scalable Storage Architecture to Support Service Storage and Service Discovery

In the service market, a centralized service registry maintains a record for each registered service, which contains various types of pertinent information such as the corresponding service provider and QoS attributes. Such information enables the published service to be able to be discovered and invoked by a service composition protocol. However, the proliferation of services and their diverse realizations require the service registry to be able to store a large amount of service-related information. The related service discovery process also becomes a challenging task due to the existence of many candidates for a single service request. These issue has not been fully explored in the current service registry design [18][19]. In our previous work [7], we characterize the services through quantifiable features, and classify services into separate clusters based on pattern recognition algorithms [20]. This service clustering approach leads to a hierarchical service management scheme to support the large scale service solution design. Based on feature-based service characterization, we can encode both the registered services and incoming service requests as bit strings, and thus transform the service discovery problem into the string matching problem. This transformation enables us to use well-established algorithms in this field [21][22]. Furthermore, we propose a service storage architecture based on efficient yet widely-applicable data structures to support the storage and service discovery process. The main contents of this chapter have been published in [8].

3.1 Quantifying Services With Features

Each service is characterized by a collection of features with a wide range of attributes spanning from business oriented ones to system centric ones. These features belong to two basic categories:

Functional features: these features are used to describe the basic functionalities of the related service, e.g., shipping service;

Non-functional features: these features are related to a variety of constraints that have to be satisfied while meeting the functional requirements.

Based on the practical service design experiences, we tend to take into account the following service features which usually play critical roles in characterizing services [23].

Service availability quantifies the probability of being able to serve a request by one or more service providers.

Service cost represents the price tag that is asked by the service provider for delivering the related service. It usually is equivalent to the sum of the operation cost of rendering the service and the net profit expected by the service provider.

Service interface defines the gateway for communicating with service consumers and other service providers, it stipulates the rules for the service functionality to be explored.

Service reliability measures the ability of a service provider to satisfactorily deliver service in compliance with the enlisted functional and non-functional requirements in a given time period.

Service response time is computed as the elapsed time between the time point of receiving a service request and the time point of fulfilling the service functionality.

Service security evaluates how the privacy of various participants are protected during the life cycle of the service provisioning process.

Every selected feature is assigned with a numerical range over which we can score a particular service. In the following example, we quantify the response time ($\mathcal{T}$) in the scale of "1-5". A service with smaller response time is assigned a higher score.

Table 3.1 Quantification procedure of service response time ($\mathcal{T}$)

Quantitative range of $\mathcal{T}$	$\mathcal{T}$ score
$\mathcal{T} \leq 25s$	5
$25s < \mathcal{T} \leq 50s$	4
$50s < \mathcal{T} \leq 75s$	3
$75s < \mathcal{T} \leq 100s$	2
$\mathcal{T} > 100s$	1

Based on the binary notation scheme,the numerically scored service feature can be represented as a bit string. For instance, the l-th feature of $AS_{i,j}$, an atomic service of identity j and delivered by atomic service provider asp_i, is the service response time and gets a numerical score of 5 in the above quantification procedure. It is thus encoded as a binary bit string of 0101. Hereafter, we use $bs_l^{i,j}$ to denote the bit string corresponding to the l-th feature of $AS_{i,j}$. By considering all of the related features, the service is also represented as a bit string, the congregation of a collection of bit strings, each of which corresponds to a single feature.

$$AS_{i,j} \leftrightarrow bs_1^{i,j} \dots bs_l^{i,j} \dots bs_n^{i,j} \tag{3.1}$$

In Equation.3.1, the whole string $bs_1^{i,j} \dots bs_l^{i,j} \dots bs_n^{i,j}$ is referred to as a *service parameter configuration*, which encodes every related performance metrics for service $AS_{i,j}$. Every realizable service parameter configuration is published to the service registry as a candidate for the service discovery process.

This feature-based bit string representation scheme works not only for the registry services, but also for the service requests. The customers' various requirements are rep-

resented as the target score for the corresponding features used to characterize services. The service discovery process is hereby transformed into a classical string matching problem which has been studied extensively [22]. The string matching-based service discovery solution requires comparing every candidate service against the request. Accordingly, the total computational cost depends on both the number of comparison operations and the performance of each string matching operation. The algorithm design for the string matching problem is beyond the research scope of this thesis. Our research focuses on the storage architecture design for the service registry that can reduce the number of comparison operations.

3.2 Feature-guided Architecture Design

In the competitive service market, the number of services in terms of offered functionalities are limited. However,the number of potential service realizations in terms of their respective service parameter configurations can be very large. This is a reasonable hypothesis because a service provider is able to provide the same type of service with different parameter configurations. For instance, a server can serve requests with different response time values. Consider a service characterized by 6 features, and each feature is scored on a scale of "1-5". Potentially, there exist 5^6 service parameter configurations for this single service. Formally, we use $\phi = \{AS_1, \ldots, AS_m\}$ to denote the set of atomic services in a competitive service market. With respect to service AS_i, the set of its corresponding service parameter configurations is referred to as a *service group* and is denoted as $\mathcal{S}_i$. The union of $\mathcal{S}_i$, i.e., $\cup_{i=1}^{i=m}\mathcal{S}_i$, is denoted as $\mathcal{G}$.

In a storage architecture, ϕ and $\mathcal{G}$ are stored separately, and the storage requirement for ϕ is less demanding than the one for $\mathcal{G}$. The mapping relationship between $AS_i \in \phi$ and $\mathcal{S}_i \subseteq \mathcal{G}$ suggests that the former can be used to index the latter. In the storage architecture design, keeping track of the mapping relationship between service and the

corresponding service group aims to support the service discovery process, which will be discussed later. Our design is a hybrid storage architecture (HSA) that is able to store both ϕ and $\mathcal{G}$ while supporting an efficient service discovery process. HSA, illustrated in Figure 3.1, is composed of an service array used to store the set ϕ, and a collection of storage units used to store the set $\mathcal{G}$.

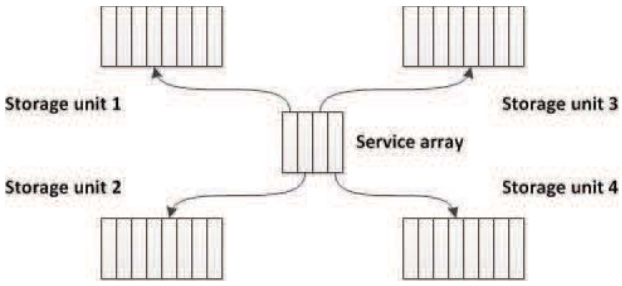

Figure 3.1 Hybrid storage architecture for service registry

Due to the limited number of services, the service array is based on the normal data structure,like array. The i-th entry of the service array stores the identity of a service, e.g., AS_i, and maintains a pointer to the storage unit i, which stores the service group $\mathcal{S}_i$. Since the size of a storage unit is usually very large due to the combinatorial parameter configurations, the focus of this section centers around the design of these storage units. In our research, we propose to use counting Bloom filter, a probabilistic data structure, to build the storage unit.

3.2.1 Standard Bloom Filters and Counting Bloom Filters

A standard Bloom filter (BF) is a hashing-based data structure representing a set of elements [24]. Compared to the hash table, BF can reduce the space requirement further and allows using simpler hash functions, which saves computational cost for lookup-intensive operations [25]. A BF is composed of a bit array denoted as $\mathbf{B}$ and g independent hash functions $\mathcal{F}_h = \{h_1(\ ), \ldots, h_g(\ )\}$. The bit array $\mathbf{B}$ is of length q and is initialized as 0 for all of the entries. Each hash function maps an element in

$\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ to an array entry. Hereafter, we use $BF^{\mathcal{A}}$ to denote the BF for set $\mathcal{A}$, and use $\mathbf{B}^{\mathcal{A}}$ to denote the related bit array. Algorithm 3.1 lists the procedures of constructing a Bloom filter representing a given set, and Figure 3.2 shows the constructed $BF^{\mathcal{A}}$ for set $\mathcal{A}$.

Algorithm 3.1 Constructing Bloom filter for set $\mathcal{A}(\|\mathcal{A}\| = n)$

```
1: for i = 0 to q do
2:    B^A[i] = 0;
3: end for
4: for i = 0 to n do
5:    for j = 0 to g do
6:       p = h_j(a_i);
7:       if B^A[p] = 0 then
8:          B^A[p] = 1;
9:       end if
10:   end for
11: end for
```

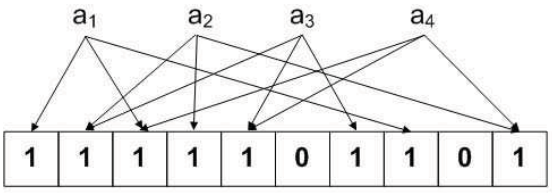

Figure 3.2 A Bloom filter example

A BF has two structural properties resulting from its construction scheme.

Property 1 : If $a \in \mathcal{A}$, then the construction process should mark every hashed position in the bit array with 1. Hence, for a given query a, if any of the hashed positions in $B^{\mathcal{A}}$ for a is found to be 0, then $a \notin \mathcal{A}$.

Property 2 : Given a query $a \notin \mathcal{A}$, it is possible to find that every hashed position of a having been marked with 1 in $B^{\mathcal{A}}$.

Given an element a and a set $\mathcal{A}$, determining whether $a \in \mathcal{A}$ is referred to as *set membership evaluation* which is used in a lot of applications [26]. A trivial approach

for the set membership evaluation is to compare each element belonging to $\mathcal{A}$ with a until we find a match, which involves a computational complexity of $\mathcal{O}(\|\mathcal{A}\|)$, where $\|\mathcal{A}\|$ represents the size of set $\mathcal{A}$. Instead of comparing each element of $\mathcal{A}$ with a, the BF-based set membership evaluation computes every hashed position of a and checks whether it has been marked with 1. According to **Property 1**, an all negative answer can be used to remove the possibility of $a \in \mathcal{A}$ with certainty. On the other hand, implied by **Property 2**, an all positive answer cannot ensure that $a \in \mathcal{A}$ due to the possible overlaps in the hashed positions among different elements belonging to $\mathcal{A}$. This issue is known as *false positive* yielded by BF-based set membership evaluation [25]. The false positive probability will decrease exponentially when g increases, which makes the influences of false positives negligible in real applications [27].

The overlaps in the hashed positions among different elements make it unable to support BF-based deleting operation on a set. For instance, deleting a_i from $\mathcal{A}$ requires the BF to reset every hashed position $\mathbf{B}^{\mathcal{A}}[h_l(a_i)] : \forall l = 1, \ldots, g$ to zero. If $h_l(a_j) = 1$ for $j \neq i$, then the deleting operation of a_i will affect the associated hashed position of another element a_j. This will result in an incorrect BF representation for a_j, whose g hashed positions should always be marked as "1". To address this issue, counting Bloom filter (CBF), an extension of the BF, is developed by setting the entry attribute as a counter value rather than a bit as in BF [28]. During the construction of a CBF, every hashed position for an element is increased by "1" rather than marked as "1". A counterpart of Figure 3.2 is shown in Figure 3.3, which illustrates the CBF representation for set $\mathcal{A}$. The value of each array entry equals the number of input arcs, i.e., the number of elements hashed into this entry. Besides the capability of supporting the deleting operation, the introduction of counter values also enables the CBF to keep track of the number of elements having been hashed into a given position. This property will be used in the latter for the CBF-based storage unit design. The set membership evaluation based on a CBF is similar to the BF: if any of the g hashed position of query a is found

to be 0 in $\mathbf{B}^{\mathcal{A}}$, then $a \notin \mathcal{A}$.

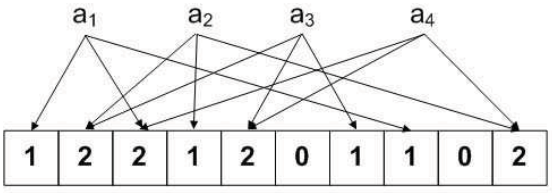

Figure 3.3 A counting Bloom filter example

3.2.2 Storage Unit Design Using Counting Bloom Filters

In this section, we will discuss the storage unit design for the service groups. Each unit stores a service group for a given service, and consists of two parts: CBF-based counter value array and a collection of linked lists. Each linked list is indexed by an array entry of CBF. Compared to the normal CBF, each entry not only stores the counter value, but also contains a pointer to the associated linked list. This storage design is based on the concept proposed in [29], which is referred to as *pruned fast Hash table* in the original paper. The major structural property of this design is the placement of element: with respect to an element and the set of corresponding hashed positions, the element is stored in the linked list whose length is the smallest according to the counter values of these hashed positions. In order to fulfill the functionality of service discovery, the HSA needs to keep track of the chaining relationships connecting service, service parameter configuration, and service provider. This motivates our extension of the classical CBF configuration: as shown in Figure. 3.4, each node in the attached list stores the bit string representing a service parameter configuration and its associated service provider. Figure. 3.4 uses the storage unit structure for service group $\mathcal{S}_1$ as an illustrative example.

$\mathcal{S}_1$ is composed of four service parameter configurations, which are denoted as AS_1^1, AS_2^1, AS_3^1 and AS_4^1 respectively. AS_2^1 is provided by SP_3, AS_3^1 is provided by SP_1. SP_2

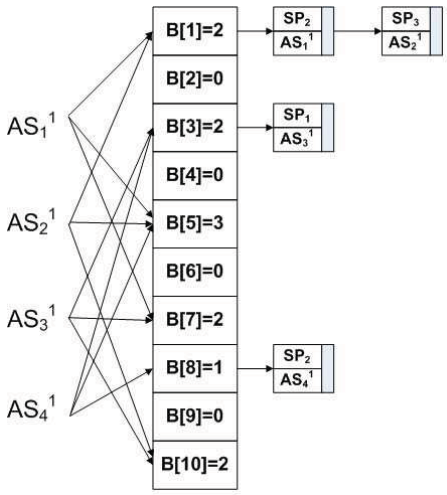

Figure 3.4 CBF-based storage unit for service group $\mathcal{S}_1$

is able to deliver two different service parameter configurations, AS_1^1 and AS_4^1. Here, with respect to AS_j^i, the superscript i is used to identify the service and corresponding service group; the subscript j denotes the j-th service parameter configuration for this service. SP_j stands for the j-th atomic service provider being able to deliver the service AS_i corresponding to $\mathcal{S}_i$. Note that we use asp_l to represent the l-th atomic service provider in Chapter 1 and 2, where l works as a global index. Here, the subscript l of SP_l can be thought of as a local index within the scope of $\mathcal{S}_i$. To avoid the confusion, we also use SP rather than asp here. In the example, there are three hashed positions for AS_3^1, which are $3, 7$ and 10. The associated counter values are $\mathbf{B}[3] = 2, \mathbf{B}[7] = 3$ and $\mathbf{B}[10] = 2$ respectively. The smallest counter value among the hashed positions of AS_3^1 is $\mathbf{B}[3]$, and thus the element is placed in the linked list pointed by $\mathbf{B}[3]$. Note that the counter value stored in an array entry does not necessarily reflect the true length of its attached linked list. For instance, $\mathbf{B}[5] = 3$, but its attached linked list is empty. Algorithm 3.2 illustrates the operation of inserting a service instance into the storage unit for its corresponding service group.

In Algorithm 3.2, i is the identity of the target storage unit for the service group $\mathcal{S}_i$

Algorithm 3.2 insertCBF($\mathcal{S}_i$, AS_j^i): Inserting a service parameter configuration into the CBF-based storage unit

1: Initialize $\mathbf{D}$ as a zero array of size g^i;
2: **for** $y = 1$ to g^i **do**
3: $\quad \mathbf{D}[y] = h_y^i(AS_j^i)$;
4: $\quad \mathbf{B}^i[\mathbf{D}[y]] = \mathbf{B}^i[p[y]] + 1$;
5: **end for**
6: $m = \mathbf{argmin}(\mathbf{B}^i[\mathbf{D}[y]]), \forall y = 1, \ldots, g^i$;
7: Attach a node consisting of AS_j^i and $SP(AS_j^i)$ to $\mathbf{list}_m$ of BF^i;

corresponding to AS_i. The CBFs corresponding to different service groups are heterogeneous in their respective parameter settings. Hence, we use g^i to represent the number of hash functions used by BF^i and h_y^i to represent its y-th hash function. The operation of line 6 is to locate the linked list with the smallest counter value among g^i hashed positions for service parameter configuration AS_j^i. At line 7, $\mathbf{list}_m$ represents the linked list pointed by the m-th array entry, and $SP(AS_j^i)$ stands for the service provider for AS_j^i, which also needs to be stored. Algorithm 3.3 lists the HSA-based service discovery protocol. In accordance with the service requirements analysis, the service discovery protocol can be separated into two steps. First, the service discovery process needs to find the service that can fulfill the functional requirements; secondly, the service discovery process needs to pinpoint the candidate service parameter configuration which can fulfill the non-functional requirements affiliated with this service request. Therefore, the HSA-based service discovery process is separated into two stages:

1. Determine the storage unit for the service group that can fulfill the functional requirements of a service request.

2. Discover the service parameter configuration that can fulfill the non-functional requirements of a service request from the storage unit obtained in the first stage.

In Algorithm 3.3, RS_f denotes the service request represented by its functional requirements. RS_{nf} denotes the bit string representing the non-functional requirements

Algorithm 3.3 hsaServiceDiscovery(RS_f, RS_{nf}): HSA-based service discovery

```
1: i = FindIndex(RS_f);
2: if i = null then
3:    print "There is no match to the service request in terms of the functional requirement."
4:    return
5: else
6:    Initialize D as a zero array of size g^i;
7:    for y = 1 to g^i do
8:       D[i] = h_y^i(RS_nf);
9:    end for
10:   m = argmin(B^i[D[y]]), ∀y = 1, ..., g^i;
11:   if D[m] = 0 then
12:      print "There is no match to the service request in terms of the non-functional requirements."
13:      return
14:   else
15:      d ← LinkedListSearch(list_m, RS_nf);
16:      if d ≠ null then
17:         print "d.sp is the service provider."
18:         return
19:      else
20:         print "There is no match to the service request in terms of the non-functional requirements."
21:         return
22:      end if
23:   end if
24: end if
```

of the service request. The operations listed between line 1 and line 5 belong to the first stage of the HSA-based service discovery protocol. As we discussed above, the i-th entry of the service array stores the identity of a service. The operation of **FindIndex** iterates over the service array and returns the service identity that can satisfy the functional requirement of the request. A returned *null* value of i indicates that there does not have a service group being able to meet the functional requirements of the request. The array of $\mathbf{D}$ stores the hashed positions for the request. In accordance with the set membership evaluation criterion of CBF, we check whether RS_{nf} can find a match in the service

group based on the smallest counter values in array **D**. These are the operations listed at lines 10 and 11. A positive answer indicates that it is possible to have a candidate service parameter configuration matching the request. This operation is a fast lookup without 100% guarantee due to the false positives of CBF. A nonzero value of $\mathbf{D}[m]$ leads to the searching operation by traversing the associated linked list. Note that the smallest element of **D** not only works for evaluating the element membership but also helps selecting the linked list to be searched. **LinkedListSearch** represents a standard searching operation over a linked list [26]. The returned node is denoted as d, and $d.sp$ stands for the service provider identity stored in d. A returned null node indicates that there is no match to the request and the original positive feedback is caused by false positives.

Algorithm 3.3 is composed of three major operations: (1) the operation of **FindIndex** determines the particular service for fulfilling the requested functional requirement, the computational cost of which depends on the number of registered services; (2) the operations at lines $6-10$ perform the set membership evaluation and chooses the linked list to be searched, the computational cost of which depends on the number of hash functions, i.e., g^i; (3) the computational cost of searching operation of **LinkedListSearch** depends on the length of the linked list bounded by $\mathbf{B}^i[\mathbf{D}[y]]$. Among these three operations, **LinkedListSearch** contributes the most to the total computational cost because neither the number of registered services nor the number of hash functions will be a big value.

3.2.3 Performance Analysis of the HSA-based Service Discovery Process

As discussed previously, the bit string-based service discovery process requires comparing a service request with every candidate service parameter configuration. The performance of this type of service discovery process is affected by two factors: the computational cost of each string matching operation and the number of comparison

operations. Our design motivation is to reduce the number of these comparison operations. The focus of this section is to quantitatively analyze the performance of the HSA-based service discovery process in terms of the expected number of comparison operations.

An inherent structural feature of the CBF-based storage unit is to distribute the candidate service parameter configurations into a set of linked lists. Moreover, the construction of each linked list is guided by the counter value stored in the array. Therefore, the second stage of the HSA-based service discovery process, which consists of a number of comparison operations, is conducted over these distributed linked lists. Let $\mathcal{C}_d$ denote the number of comparison operations required for the CBF-based storage unit design, then we have

$$\mathbf{E}[\mathcal{C}_d] = 0.5 \times \frac{\mathcal{L}_d}{2} + 0.5 \times \gamma \times \mathcal{L}_d = (0.25 + 0.5\gamma)\mathcal{L}_d \tag{3.2}$$

In Equation.3.2, $\mathcal{L}_d$ represents the expected length of the linked list for the CBF-based storage unit design and γ represents the false positive probability for the CBF-based set membership evaluation. With respect to a given service request, we assume that the probability of being able to find a match is 0.5. If a request does not have a match in the set of the candidate service parameter configurations, the CBF-based set membership evaluation should be able to block the request at the very beginning without even initializing the search operation on the linked list. However, due to the false positives of the CBF, a search operation which is certainly to fail may still be launched. This failed operation cannot find match after traversing the whole linked list. If a request does have a match in the set of the candidate service parameter configurations, an average $\mathcal{L}_d/2$ comparison operations have to be performed. This is due to the assumption that the position of that particular match in the linked list follows the uniform distribution. This analysis leads to Equation.3.2. On the other hand, without the help of CBF-guided distributed architecture design, a normal storage unit design puts all of the candidate

service parameter configurations in a single linked list. Let $\mathcal{C}_n$ denote the number of comparison operations required for the normal storage unit design, then we have

$$\mathbf{E}[\mathcal{C}_n] = 0.5 \times \frac{n}{2} + 0.5 \times n = 0.75n \tag{3.3}$$

The analysis for the normal storage unit design is similar to the above. Here, the length of the linked list is n, the number of candidate service parameter configurations. Moreover, without the help of the CBF-based set membership evaluation, all of the requests which do not have a match in the set of the candidate service parameter configurations, have to traverse the whole linked list before being able to find that there does not exist a match. Equation.3.2 differs with Equation.3.3 on two parts: $\mathcal{L}_d$ versus n; and 0.5γ versus 0.5. The first part comes from the fact that the CBF-based storage unit provides a distributed structure. The second part comes from the CBF-based set membership evaluation. Both parts contribute to make $\mathbf{E}[\mathcal{C}_d]$ being smaller than $\mathbf{E}[\mathcal{C}_n]$.

3.2.4 An Analytical Performance Metrics to Evaluate the HSA-based Service Discovery Process

In the above, we discuss the difference between $\mathbf{E}[\mathcal{C}_d]$ and $\mathbf{E}[\mathcal{C}_n]$, which directly depends on the expected length of the linked list to be searched, $\mathcal{L}_d$. However, this variable cannot be analytically formulated and requires numerical simulation when adopting it to evaluate the system performance. In this section, we will investigate an analytically solvable metrics, which can enable the system performance to be predicted without running experimental simulations.

As shown in Algorithm 3.3, if every hashed position is discovered to be nonempty, then we will search the linked list pointed by the entry whose counter value is the smallest among all of the hashed positions. Accordingly, the number of comparison operations invoked in our approach is bounded by the counter values. Hence, we use the expected counter value as the evaluation metrics. In the following, we illustrate how to

get an analytically solvable formula for the expected counter value based on probabilistic analysis. Let V denote the counter value, we have

$$\mathbf{E}[V] = \mathbf{E}[\mathbf{E}[V|\tilde{g} = i]]. \tag{3.4}$$

Equation.3.4 computes $\mathbf{E}[V]$ using iterated expectation. $\mathbf{E}[V|\tilde{g} = i]$ represents the conditional expectation of $\mathbf{E}[V]$ when there have i different hashed positions. It is computed as ng/i. Here, n represents the number of candidate service parameter configurations. We use $\tilde{g}$ to denote the number of different hashed positions in order to differentiate it with g, i.e., the number of hash functions to be used. Therefore, $\mathbf{E}[V]$ is computed as follows:

$$\begin{aligned} \mathbf{E}[V] &= \mathbf{E}[\mathbf{E}[V|\tilde{g} = i]] \\ &= \sum_{i=1}^{g} \mathbf{E}[V|\tilde{g} = i]\mathbf{Pr}(\tilde{g} = i) \\ &= \sum_{i=1}^{g} \frac{ng}{i}\mathbf{Pr}(\tilde{g} = i) \end{aligned} \tag{3.5}$$

In Equation.3.5, $\mathbf{Pr}(\tilde{g} = i)$ represents the probability that g hashed functions produce i different positions, which is computed in Equation.3.6.

$$\mathbf{Pr}(\tilde{g} = i) = \binom{q}{i} \frac{\sum_{a=0}^{i}(-1)^a\binom{i}{a}(i-a)^g}{q^g} \tag{3.6}$$

In Equation.3.6, the item of $\binom{q}{i}$ represents the number of combinations of choosing i positions from q available entries; the item of $\sum_{a=0}^{i}(-1)^a\binom{i}{a}(i-a)^g$ represents the number of permutations of putting g hashed results into i positions and none of these i positions is empty, which is derived based on the *generating function for permutation* [30]; the item of q^g represents the number of combinations of g hashed positions. In [29], $\mathbf{Pr}(\tilde{g} = i)$ is derived based on the method of induction. In this research, we derive $\mathbf{Pr}(\tilde{g} = i)$ from a different perspective and obtain a more compact analytical formula shown in Equation.3.6. Substituting Equation.3.6 into Equation.3.5, we have

$$\mathbf{E}[V] = \sum_{i=1}^{g} \frac{ng}{i}\binom{q}{i} \frac{\sum_{a=0}^{i}(-1)^a\binom{i}{a}(i-a)^g}{q^g} \tag{3.7}$$

Equation.3.7 indicates that $\mathbf{E}[V]$ is a function of n and g which are the system parameters of the CBF. Hence,$\mathbf{E}[V]$ functions as an performance metrics that can be computed beforehand and is well suited to predict the system performance. In the following, we conduct a simulation-based experimental study to compare the expected counter value with the expected length of the linked list, which is directly related to the number of comparison operations.

3.2.5 Experimental Study

In this section, we will investigate the relationship between the expected counter value and the expected length of the linked list to be searched. In the experiment, we consider a variety of parameter settings for the CBF along with two different numbers of service parameter configurations. For illustration purposes, we report both the expected counter value and the expected length of the linked list to be searched.

Table 3.2 20000 candidate service parameter configurations

Expected counter value					
	q = 90	q = 95	q = 100	q=105	q=110
g = 4	848	794	761	729	695
g = 6	1269	1211	1120	1099	1036
g = 8	1688	1603	1515	1427	1367
Expected length of the linked list to be traversed					
	q = 90	q = 95	q = 100	q=105	q=110
g = 4	493	459	442	438	431
g = 6	672	673	583	563	558
g = 8	887	851	804	806	740

Table.3.2 reports a case where there have 20000 candidate service parameter configurations. The upper part of the table reports the expected counter value computed by Equation.3.7 for different parameter settings. The lower part of the table reports the expected length of the linked list to be searched. In Table.3.3, we report a case where there have 40000 candidate service parameter configurations, and the applied parameter

settings are the same as in Table.3.2.

Table 3.3 40000 candidate service parameter configurations

Expected counter value					
	q = 90	q = 95	q = 100	q=105	q=110
g = 4	1725	1598	1540	1471	1407
g = 6	2562	2449	2303	2204	2109
g = 8	3352	3229	3092	2867	2776
Expected length of the linked list to be traversed					
	q = 90	q = 95	q = 100	q=105	q=110
g = 4	997	968	842	873	825
g = 6	1343	1323	1285	1105	1064
g = 8	1707	1704	1653	1587	1437

As shown in Table 3.2 and Table 3.3, the expected length of the linked list for every system parameter setting is much smaller than n, i.e., $\mathcal{L}_d \ll n$. This fact helps supporting the claim that $\mathbf{E}[\mathcal{C}_d] \ll \mathbf{E}[\mathcal{C}_n]$. The experimental results also show that every expected length of the linked list is smaller than the corresponding expected counter value. In some cases, the difference is large. Therefore, the expected counter value should be treated as a conservative yet applicable approach to estimate the expected length of the linked list to be searched, which directly controls the number of comparison operations. Moreover, although the expected counter value is larger than the expect length of the linked list, it is still smaller than n. Therefore, the analytical formula for predicting the system performance, which is based on the expected counter value, keeps track of the difference between the two storage designs in terms of the expected number of comparison operations.

Based on the results reported in Table 3.2 and Table 3.3, a larger size of counter array for the CBF, i.e., q, reduces both the expected counter value and the expected length of the linked list to be searched. This is due to the fact that a larger counter array allows more linked lists to be built, which leads to a more distributed storage configuration. Although the total storage requirement is still the same, the expected

number of comparison operations can be reduced accordingly. In practice, the counter array of CBF can be maintained in a high-bandwidth and small on-chip memory [29]. This makes the computation of hash functions very fast, which enables the composite service provider to block the request very quickly if it gets a negative answer from the CBF-based set membership evaluation. This feature also helps the customer to decide whether to modify the request or just quit the market in a very short time period, which is a desirable characteristic in a competitive business environment. On the other hand, the size of on-chip memory limits the size of counter array that we can use. Therefore, it is not feasible to keep increasing the size of counter array for CBF without limitation.

The construction scheme of CBF results in the increments of the counter values when we use more hash functions. This fact is reflected by the results shown in both tables. Nevertheless, Bloom filter theory suggests that the false positive probability decreases exponentially when we increase the number of hash functions. According to Algorithm 3.3, the HSA-based service discovery protocol decides whether to block the request based on the CBF-based set membership evaluation. A small false positive probability is always desirable for this kind of functionality. Therefore, choosing the number of hash functions should be treated as a design trade-off that balances different factors.

3.3 Summary and Discussion

From the perspective of service science and engineering, our contributions in this chapter are threefold. First, after discussing our previous work on feature-based service quantification, we distinguish between the concept of service and the concept of service parameter configuration in terms of functional requirements and non-functional requirements. Further, we design a hybrid storage architecture using counting Bloom filter. Rather than focusing on the storage solution only, we also formalize the service

discovery protocol based on the proposed architecture.

The proposed storage architecture still has several limitations that need to be lifted in the future research. Regarding the HSA, the counter values and the associated linked lists for each storage unit are configured during the CBF building process based on the existing service group. If a service parameter configuration is newly created and is required to be added into the related storage unit, then the CBF and its related linked lists may have to be rebuilt in order to allow the service discovery protocol perform correctly. This rebuilding operation will cost a lot of computational resources if there are frequent service insertions. An incremental insertion algorithm, proposed by Song et.al [31], can be applied to support dynamic element insertion. Another approach is to design storage unit based on dynamic Bloom filter, which supports the dynamic set management and aims to control the false positive probability even the set size increases [32].

In [7], we develop a pattern recognition-based service clustering scheme to organize the services in a hierarchical structure to accelerate the service discovery process. This technique can be used to refine the HSA design. Instead of using a single CBF-based storage unit to store the whole service group corresponding to a given service, we can build a set of storage units with each of which storing a service cluster decomposed from the service group. Hence, the service discovery process will be performed on a smaller sized storage unit. This increased granularity can improve the performance of service discovery by reducing the number of comparison operations further. The core idea underlying this service clustering-based HSA design is to build a more distributed storage structure compared with the current HSA. However, the chaining relationships connecting service, service group and service cluster need to be carefully maintained in order to support the indexing and discovery operations.

CHAPTER 4. Design Service Level Agreement for Competitive Service Market

In the competitive service market, the business relationships between a composite service provider and its controlled atomic service providers are framed in terms of the service level agreements (SLAs). Every composite service provider signs a SLA with each of its controlled atomic service provider. Without signing SLA with an atomic service provider, the composite service provider cannot include the related atomic service in the service composition process.

From the system management perspective, a SLA provides parameters that a system manager can use to discover, invoke and monitor various types of resources during the service provisioning process. In a variety of application domains, such as wireless network management and multimedia content delivery, SLA has been used to deliver services that both meet the customer's QoS requirements and satisfy the system resource constraints [33][34][35]. In the field of services computing, a SLA-based resource management system for the enterprise information architecture is investigated in [36]. In this research, the authors propose a queuing-based performance prediction scheme to support the service selection while ensuring that the realized QoS metrics match the pre-determined SLA specifications.

Our research focus is on the SLA design in the context of competitive service market. To the best of our knowledge, most of the existing works on the SLA design is to model the SLA establishment as a negotiation process [37][38][39]. Due to the complex business

dynamics of a competitive service market, the negotiation-based SLA design protocol cannot capture every possible scenario during the life-cycle of a service market. We will address this issue by identifying different SLA design patterns and investigate their respective design methodologies.

We start this chapter by introducing the constituent components of the SLA. Secondly, we identify two essential SLA design patterns in the context of competitive service market and discuss their respective design methodologies. Thirdly, we build a stochastic model for the dynamic business relationships framed by coexisting SLA design protocols and derive a set of performance metrics that can help market players to make strategic and operational decisions.

4.1 Service Level Agreement: the Constituent Components and the Basic Design Patterns

A generic SLA structure is specified in terms of three components: service identity, QoS metrics and service cost [40][41][42]. In the current research, we focus on the SLA design issues between a composite service provider and its controlled atomic service providers. Hence, the service identity,e.g., AS_i stands for the atomic service delivered by the related atomic service provider. Each service is associated with a *QoS metrics set* developed from various evaluation perspectives, such as availability, response time and security level,etc. The QoS metrics set for a given service AS_i is denoted as $\mathcal{Q}_i = \{Q_1, \ldots, Q_n\}$, where n represents the number of QoS metrics to be considered and Q_j denotes the j-th QoS metrics. In the competitive service market, each service can have multiple providers, which are heterogeneous in terms of the realization of each QoS metrics. A particular service provider asp_t of AS_i assigns a specific value to each metrics, e.g., Q_j. This assignment is based on the capability of service provider itself and its coordination with the service purchaser. For instance, a data center determines

its average response time based on the combined hardware software capability and the predicted incoming rates of the requests. We use $\mathcal{QV}_i^t = \{QV_{i,1}^t, \ldots, QV_{i,n}^t\}$ to store these values, where $QV_{i,j}^t$ is the value for the j-th QoS metrics, Q_j assigned by asp_t with respect to service AS_i. $\mathcal{QV}_i^t$ is referred to as *QoS value set*. A signed SLA instance is therefore formulated as a triple $SLA = (AS_i, \mathcal{QV}_i^t, \mathcal{P}_i)$, where AS_i indicates the service to be delivered. The signed SLA obligates the involved atomic service provider to deliver the service in accordance with the stipulated QoS value set, i.e., $\mathcal{QV}_i^t$, while the composite service provider pays the listed service charge, i.e., $\mathcal{P}_i$. Naturally, the service charge should cover the operation cost of atomic service provider while generating a certain amount of profit to itself.

4.1.1 The SLA Design Patterns

Bilateral negotiation is one of the most commonly used decision making processes in the business world and fits to the SLA design scenario where there only have one service seller and one service purchaser [43]. Figure.4.1 displays the bilateral negotiation pattern for the SLA design. The arrows at both ends reflect the dynamics of offer and counter offer throughout the bilateral negotiation process.

Figure 4.1 Bilateral negotiation pattern of SLA design

Figure.4.2 is the activity diagram of the bilateral negotiation process [44]. Besides the essential decision making processes made on both the seller side and the purchaser side, we also factor in the negotiation deadline. This is due to the reason that the negotiation will not continue forever in practice. When implementing a negotiation

protocol, this type of negotiation deadline is instantiated as the maximal number of negotiation iterations.

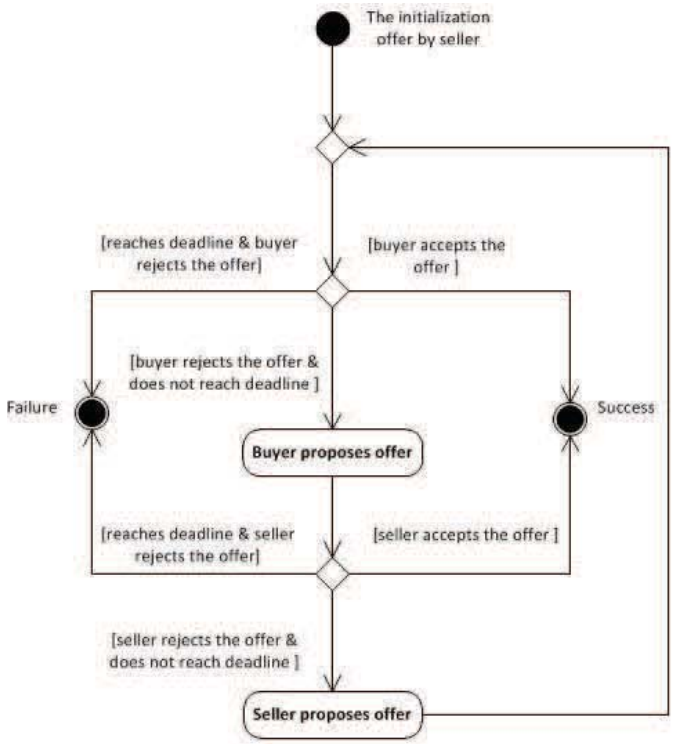

Figure 4.2 Activity diagram for the bilateral negotiation-based SLA design

A basic characteristic of bilateral negotiation process is that the reached agreement only reflects a compromised result after several rounds of bargaining. It is ad-hoc and very hard to be predicted. In terms of the SLA design in the competitive service market, this type of uncertainty indicates that the service cost or the QoS value set cannot be determined beforehand. Under some scenarios, it is necessary for the composite service provider to reach a target for a specific QoS metrics over a particular service. For instance, composite service provider csp_o needs to ensure that at least one of its controlled atomic service providers is able to deliver an atomic service with the mean time between failure (MTBF) being longer than two hours. We term the related design problem as *target-oriented SLA design*, which is the second type of SLA design pattern

to be discussed in this research. In order to ensure signing a SLA with a predetermined target, having more than one candidate service providers is preferable. Comparing with the bilateral negotiation-based SLA design featured as a one-to-one business relationship, the target-oriented SLA design tends to involve multiple service providers. The auction process fits naturally to this type of business scenario with twofold features: one-to-many relationship and a predetermined target need to be reached [45].

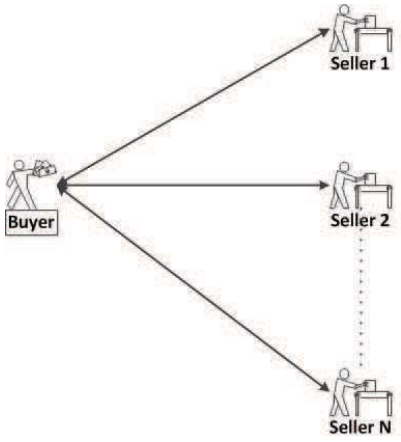

Figure 4.3 An example of buyer-dominant auction for the SLA design

Figure.4.3 is a conceptual diagram of the auction process, where we have one service purchaser which stands for the composite service provider in the competitive service market, and multiple atomic service providers.

Figure.4.4 is the activity diagram for the auction process. A synchronization transition is included to ensure every invited seller will be considered in the auction protocol. Introducing this mechanism is important due to the possible communication delay in the real service market.

From the viewpoint of economic theory, the difference between the bilateral negotiation and the auction is far from clear [46]. For instance, a one-to-many auction process can be simulated using multiple one-to-one bilateral negotiation processes. The industrial practices and empirical evidence show that the bilateral negotiation can be used to

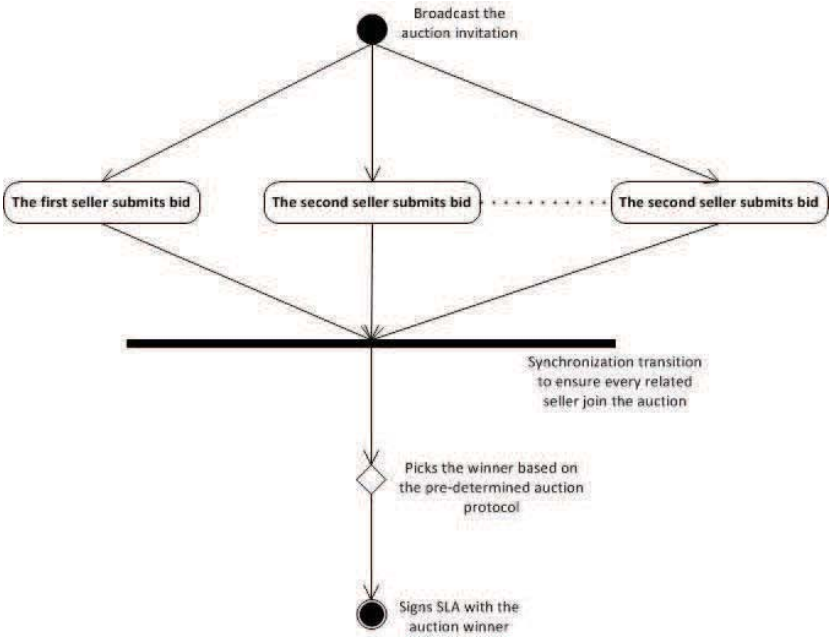

Figure 4.4 Activity diagram for the target-oriented SLA design pattern using auction

establish very complicated business relationships, which usually involve the bargaining, estimation and concession. On the other side, the auction process performs well to meet a well-established business target by exploiting potential competitions.

4.2 Negotiation-based Multi-attributes SLA Design Using Bayes Estimator

In the practical SLA design, price and quality-of-service (QoS) are two essential components that need to be considered. As we discussed previously, QoS is composed of a set of performance metrics. In practice, this set of metrics can be aggregated into a quality level ($\mathcal{L}$) which is treated as an umbrella QoS index dedicated to capture the whole set of related performance metrics in a unified way. A typical formulation of

quality level is given in Equation.4.1.

$$\mathcal{L} = \sum_{i=1}^{i=n} w_i q_i \tag{4.1}$$

In Equation.4.1, w_i stands for the weight of the corresponding QoS metric q_i. The exact formula of Equation.4.1 and the weight values can vary by the studied system. In this research, we just assume the existence of aggregated quality level. A SLA design taking both price tag and quality level into account is referred to as a multi-attributes SLA design. In this section, we are going to discuss the concrete design issues for the negotiation-based multi-attributes SLA design using Bayes estimator.

Performing as autonomous agents in the competitive service market, both composite service providers and atomic service providers hold private knowledge over their respective business goals, operation costs and even social responsibilities. This knowledge comes from the understanding of its own system dynamics and the accumulated experiences learned from interacting with other agents. For example, from the profit-oriented perspective, a higher selling price indicates a larger profit margin to an atomic service provider. Nevertheless, a higher selling price will also reject the potential service purchasers and cause future losses. Delivering service with higher quality level can attract the potential service consumer but the associated operation cost may counteract those positive effects. In the real business world, there always exists a complicated balance point between increasing and decreasing in terms of both price and quality level. This is especially true when we not only consider an agent's own business interest but also take into account its relationships with other market players. Based on this knowledge, an agent should be able to make an evaluation about how to maximize its business benefit from doing business with other agents. In terms of the price and quality level which characterize the business relationship connecting a composite service provider and an atomic service provider, this evaluation is represented as an *optimal utility pair*:

- With respect to a given atomic service, a composite service provider holds a pair

of quality level and price, which can return itself the maximal business benefit by purchasing this service from the atomic service provider.

- With respect to a given atomic service, an atomic service provider holds a pair of quality level and price, which can return itself the maximal business benefit by selling this service to the composite service provider.

For notation purposes, the optimal utility pair for an atomic service provider, asp_i, is denoted as $\theta^i = (\theta^i_p, \theta^i_l)$, where θ^i_p and θ^i_l represent the service price and the quality level in an optimal SLA configuration from the asp_i's perspective. Similarly, the optimal utility pair for a composite service provider,$\theta^o = (\theta^o_p, \theta^o_l)$, where θ^o_p and θ^o_l represent the service price and the quality level in an optimal SLA configuration from the csp_o's perspective. In a competitive service market, this type of information tends to be hidden from other agents.

Naturally, both a composite service provider and an atomic service provider would like to sign a SLA with specifications being near to their respective optimal utility pairs. Therefore, in the context of SLA design, the bilateral negotiation process conducted between a composite service provider and an atomic service provider is characterized by a process of making concessions from the optimal utility pairs on both sides. From the viewpoint of a negotiation agent, if the optimal utility pair of the negotiation counterpart is known, then it can develop a strategy to maximize its utility. From the game theoretic perspective, this is to transfer a game of incomplete information to a game of complete information.

In our research, we assume the accessibility of public information resources, which enables an agent to build a subjective evaluation of its negotiation counterpart. Furthermore, the offer proposed by the negotiation counterpart constitutes additional information that can be exploited. The combination of pre-owned subjective evaluation and updated information can be efficiently utilized under the framework of Bayesian analy-

sis [47][48]. We formulate the SLA design scheme between csp_o and asp_i in Algorithm.4.1 which includes a Bayes estimator in each negotiation iteration.

Algorithm 4.1 Bayes estimator-based SLA design

1: $l = 1$, $\eta = 0$;
2: Initialize $\widehat{\theta_o}$;
3: **while** $(k \leq \mathcal{K})$ and $(\eta = 0)$ **do**
4: $\alpha = \mathbf{calOffer_i}(\theta_i, \widehat{\theta_o})$
asp_i informs csp_o of its offer: α;
5: $\eta = \mathbf{evaOffer}_o(\theta^o, \alpha)$;
6: **if** $\eta = 1$ **then**
7: **Return** (SLA is signed in terms of α);
8: **end if**
9: $\widehat{\theta^i} = \mathbf{estOUP}(\alpha)$;
10: $\beta = \mathbf{calOffer_o}(\theta^o, \widehat{\theta^i})$
csp_o informs asp_i of its counter offer: β;
11: $\eta = \mathbf{evaOffer}_i(\theta^i, \beta)$;
12: **if** $\eta = 1$ **then**
13: **Return** (SLA is signed in terms of β);
14: **end if**
15: $\widehat{\theta^o} = \mathbf{estOUP}(\beta)$;
16: $k = k + 1$;
17: **end while**
18: **if** $\{\eta = 0\}$ **then**
19: asp_i does not sign SLA with csp_o;
20: **end if**

In Algorithm 4.1, lines between 4 and 16 represent the negotiation conducted at each iteration. In reality, the negotiation cannot continue forever. Hence, we let $\mathcal{K}$ denote the maximum acceptable rounds of negotiation. η is a control variable monitoring the negotiation status. Once an agreement is reached, η will be switched to 1 and the algorithm will jump out of the *while* loop. If both agents cannot reach an agreement within the maximal number of iterations, then no SLA will be signed. At each iteration, each negotiation agent has two essential operations:

- $\mathbf{evaOffer}_i$ for asp_i and $\mathbf{evaOffer}_o$ for csp_o. This set of operations is to evaluate whether accept the counterpart's offer or not. For instance, if the counterpart's

offer lies within a certain range, then it will accept the offer. The implementation of $\mathbf{evaOffer}_i$ can differ with the implementation of $\mathbf{evaOffer}_o$ due to their respective utilities.

- $\mathbf{estOUP}_i$ for asp_i and $\mathbf{estOUP}_o$ for csp_o. This set of operations is to estimate the optimal utility pair of the negotiation counterpart.
- $\mathbf{calOffer}_i$ for asp_i and $\mathbf{calOffer}_o$ for csp_o. This set of operations is to calculate the offer based on a given strategy.

The operations of $\mathbf{evaOffer}_{i(o)}$ and $\mathbf{calOffer}_{i(o)}$ will be discussed in more details later. Since Algorithm.4.1 assumes that it is asp_i to initialize the negotiation, and thus the estimation of θ^o has to be manually setup at the beginning of the first iteration. This is what line 2 represents and can be calculated using the prior subjective estimation of θ^o by asp_i. The core of Algorithm.4.1 is the Bayes estimators: $\mathbf{estOUP}_i$ and $\mathbf{estOUP}_o$. In the following, we discuss how the Bayes estimator is derived in the context of bilateral negotiation.

Due to the symmetrical format, we just look into the estimation of θ^i from the perspective of csp_o. The estimation quality is evaluated based on a selected criterion, like the quadratic loss function, i.e., $\mathrm{L}(\widehat{\theta^i}, \theta^i) = (\widehat{\theta^i} - \theta^i)^2$. Here, $\widehat{\theta^i}$ denotes the computed value based on Bayes estimator. It should be able to minimize the expected value of $\mathrm{L}(\widehat{\theta^i}, \theta^i)$ given updated information: the offer of asp_i, i.e., α.

$$\begin{aligned} \widehat{\theta^i} &= \mathbf{argmin}\ \mathbf{E}[\mathrm{L}(\widehat{\theta^i}, \theta^i)|\alpha] \\ &= \mathbf{argmin}\ \mathbf{E}[(\widehat{\theta^i} - \theta^i)^2|\alpha] \quad \forall \widehat{\theta^i} \in \Phi \end{aligned} \tag{4.2}$$

By re-writing $\widehat{\theta^i} - \theta^i$ as $\widehat{\theta^i} + \mathbf{E}[\theta^i|\alpha] - \mathbf{E}[\theta^i|\alpha] - \theta^i$, we have

$$\mathbf{E}[(\widehat{\theta^i} - \theta^i)^2|\alpha] = y_1 + y_2 + y_3 \tag{4.3}$$

where

$$y_1 = \mathbf{E}[(\widehat{\theta^i} - \mathbf{E}[\theta^i|\alpha])^2|\alpha]$$

$$y_2 = \mathbf{E}[(\mathbf{E}[\theta^i|\alpha] - \theta^i)^2|\alpha]$$

$$y_3 = 2\mathbf{E}[(\mathbf{E}[\theta^i|\alpha] - \theta^i)(\widehat{\theta^i} - \mathbf{E}[\theta^i|\alpha])|\alpha]$$

Note that $\widehat{\theta^i}$ does not have any control over the computation of item y_2, and $y_3 = 0$ since

$$\mathbf{E}[(\mathbf{E}[\theta^i|\alpha] - \widehat{\theta^i})(\theta^i - \mathbf{E}[\theta^i|\alpha])|\alpha] = 0.$$

Thus,

$$\begin{aligned}
\widehat{\theta^i} &= \textbf{argmin}\ \ \mathbf{E}[(\widehat{\theta^i} - \theta^i)^2|\alpha] \\
&= \textbf{argmin}\ \ \mathbf{E}[(\widehat{\theta^i} - \mathbf{E}[\theta^i|\alpha])^2|\alpha] \\
&= \mathbf{E}[\theta^i|\alpha] \\
&= \int_{\Phi} \theta^i \mathbf{f}(\theta^i|\alpha)\ \mathbf{d}\theta^i
\end{aligned} \tag{4.4}$$

Equation.4.4 explicitly stipulates how a Bayes estimator of the negotiation counterpart's optimal utility pair should be computed at each iteration. In the probability theory, this formula is also referred to as the conditional expectation [49] In Equation.4.4, $\mathbf{f}(\theta^i|\alpha)$ is the conditional distribution of θ^i given α, and is referred to as the posterior distribution in the Bayesian analysis [50]. It depends on the likelihood of α given θ^i and the prior distribution of θ^i. The formula given in Equation.4.4 is defined over a continuous domain, which can become unsolvable once the related probability distributions become complicated. In the next, we will discuss how to solve it numerically in the two-dimensional space for a multi-attributes SLA design.

4.2.1 Numerical Scheme for the Two-Dimensional SLA Design Using Bayes Estimator

The Bayes formula can become unsolvable for high-dimensional parameter space and complicated probability distributions [50]. Hence, we choose to discretize the original continuous parameter space. With respect to the two-dimensional domain of {price} ×

{quality level}, the discretization operation is formally represented as:

$$\begin{aligned}(\theta_p^{i,a}, \theta_l^{i,b}, a = 1, ..., n_1, b = 1, ..., m_1) &= \textbf{discreteSpace}([\theta_{p.low}^i, \theta_{p.high}^i] \otimes [\theta_{l.low}^i, \theta_{l.low}^i]) \\ (\theta_p^{o,a}, \theta_l^{o,b}, a = 1, ..., n_2, b = 1, ..., m_2) &= \textbf{discreteSpace}([\theta_{p.low}^o, \theta_{p.high}^o] \otimes [\theta_{l.low}^o, \theta_{l.high}^o])\end{aligned} \tag{4.5}$$

In Equation.4.5, $\theta_{p.low}^i$ and $\theta_{p.high}^i$ determine the range of θ_p^i and are held by csp_o as a prior knowledge over asp_i. Similar explanations apply to $[\theta_{l.low}^i, \theta_{l.high}^i]$, $[\theta_{p.low}^o, \theta_{p.high}^o]$ and $[\theta_{l.low}^o, \theta_{l.high}^o]$ as well. However, the latter two reflect the prior knowledge of asp_i over csp_o. The procedure of **discreteSpace** is to build a two-dimensional grid and use the central point on each grid cell as one hypothetical optimal utility pair. With respect to this two-dimensional parameter space, we use $n_j : j = 1, 2$ to represent the number of cells along the dimension of "price" and use $m_j : j = 1, 2$ to represent the number of cells along the dimension of "quality level". Hence, calling Equation.4.5 returns two two-dimensional arrays: $(\theta_p^{i,a}, \theta_l^{i,b}, a = 1, ..., n_1, b = 1, ..., m_1)$ and $(\theta_p^{o,a}, \theta_l^{o,b}, a = 1, ..., n_2, b = 1, ..., m_2)$. The focus of this section is to discuss how the Bayes estimator is implemented on this two-dimensional parameter space and is applied to the multi-attributes SLA design which takes both price and quality level into account. The concrete design of Algorithm.4.1 in the context of multi-attributes SLA design over a discrete parameter space is given in Algorithm.4.2.

Algorithm 4.2 Two dimensional SLA design using Bayes estimator

1: $(\theta_p^{i,a}, \theta_l^{i,b}, a = 1...n_1, b = 1...m_1) = \textbf{discreteSpace}([\theta^i_{p.low}, \theta^i_{p.high}] \otimes [\theta^i_{l.low}, \theta^i_{l.low}])$;
2: $(\theta_p^{o,a}, \theta_l^{o,b}, a = 1...n_2, b = 1...m_2) = \textbf{discreteSpace}([\theta^o_{p.low}, \theta^o_{p.high}] \otimes [\theta^o_{l.low}, \theta^o_{l.high}])$;
3: $\mathbf{f}_{prior}(\theta_p^{i,a}, \theta_l^{i,b}) = \frac{1}{n_1 * m_1} \quad \forall a = 1...n_1, b = 1...m1$;
4: $\mathbf{f}_{prior}(\theta_p^{o,a}, \theta_l^{o,b}) = \frac{1}{n_2 * m_2} \quad \forall a = 1...n_2, b = 1...m2$;
5: $\widehat{\theta^o} = (\widehat{\theta^o_p}, \widehat{\theta^o_l}) = (0.5 * (\theta^o_{p.low} + \theta^o_{p.high}), 0.5 * (\theta^o_{l.low} + \theta^o_{l.high}))$;
6: $k = 1, \eta = 0$;
7: **while** $\{k \leq \mathcal{K})$ and $(\eta = 0)\}$ **do**
8: $\alpha^k = \textbf{calOffer}_i(\theta^i, \widehat{\theta^o})$;
asp_i informs csp_o of its offer: α^k;
9: $\eta = \textbf{evaOffer}_o(\theta^o, \alpha^k)$;
10: **if** $\eta = 1$ **then**
11: **Return** (SLA is signed in terms of α^k);
12: **end if**
13: $[\{\mathbf{f}^{k+1}_{prior}(\theta_p^{i,a}, \theta_l^{i,b}) : a = 1...n_1, b = 1...m_1\}, \widehat{\theta^i}] = \textbf{estOUP}_o(\alpha^k, \{\mathbf{f}_{prior}(\theta_p^{i,a}, \theta_l^{i,b}) : a = 1...n_1, b = 1...m_1\}, k)$;
14: $\beta^k = \textbf{calOffer}_o(\theta^o, \widehat{\theta^i})$;
csp_o informs the asp_i of its counter offer: β^k;
15: $\eta = \textbf{evaOffer}_i(\theta^i, \beta^k)$;
16: **if** $\eta = 1$ **then**
17: **Return** (SLA is signed in terms of α^k);
18: **end if**
19: $[\{\mathbf{f}^{k+1}_{prior}(\theta_p^{o,a}, \theta_l^{i,b}) : a = 1...n_2, b = 1...m_2\}, \widehat{\theta^o}] = \textbf{estOUP}_i(\beta^k, \{\mathbf{f}_{prior}(\theta_p^{o,a}, \theta_l^{i,b}) : a = 1...n_2, b = 1...m_2\}, k)$;
20: $k = k + 1$;
21: **end while**
22: **if** $\eta = 0$ **then**
23: asp_i does not sign SLA with csp_o;
24: **end if**

In Algorithm.4.2, the implementation issues of $\mathbf{evaOffer}_o$ and $\mathbf{evaOffer}_i$ will be discussed in section.4.2.2. An estimation-based offer proposition approach will be investigated in section.4.2.3. The Bayes estimator of the counterpart's optimal utility pair, e.g., $\mathbf{estOUP}_i$, is the core of this algorithm and is given in Algorithm.4.3.

4.2.2 Offer Evaluation Criteria

The operation of offer evaluation is to determine whether to accept or reject the negotiation counterpart's offer based on a set of criteria. We use different subscripts, i and o, in the $\mathbf{evaOffer}_i$ and $\mathbf{evaOffer}_o$ to stress that the two sides can differ in their respective evaluation criteria. In this study, we consider two types of evaluation criteria. One tends to focus on the business interest of one side only and the other one aims to improve the collaboration of two sides.

Evaluation criterion 1 The agreement will be reached if either of the following two conditions is satisfied.

1. On the side of csp_o :, If the difference between the asp_i's offer and the csp_o's optimal utility pair is within a given range, then csp_o will accept the offer. Formally, we have

$$\alpha_p^k < r_1 \times \theta_p^o \text{ and } \alpha_l^k > r_2 \times \theta_l^o \quad \exists 1 \leq k \leq \mathcal{K} \tag{4.6}$$

   In Equation.4.6, α_p^k and α_l^k represent the price offer and the quality level offer respectively which are proposed by asp_i at the k-th iteration. r_1 and r_2 reflect the concession degrees that the csp_o would like to make in the reached agreement. Naturally, $r_1 > 1$ and $r_2 < 1$.

2. On the side of asp_i : If the difference between the csp_o's offer and the asp_i's optimal utility pair is within a given range, then asp_i will accept the offer.

Formally, we have

$$\beta_p^k > r_3 \times \theta_p^i \text{ and } \beta_l^k < r_4 \times \theta_l^i \quad \exists 1 \leq k \leq \mathcal{K} \tag{4.7}$$

In Equation.4.7, r_3 and r_4 reflect the degrees of concession that the seller would like to make in the reached agreement. Naturally, $r_3 < 1$ and $r_4 > 1$.

Evaluation criterion 2 From the viewpoint of the collaboration, a settled agreement should consider the utility of both sides at the same time. However, **Evaluation criterion 1** does not take the combined utility into account. An applicable metrics to consider both sides' benefits is referred to as the aggregated utility function [37].

$$\begin{aligned} \mathcal{U}_p^{sla} &= w_p^i * \theta_p^i + w_p^o * \theta_p^o \qquad w_p^i + w_p^o = 1 \\ \mathcal{U}_l^{sla} &= w_l^i * \theta_l^i + w_l^o * \theta_l^o \qquad w_l^i + w_l^o = 1 \end{aligned} \tag{4.8}$$

In Equation.4.8, we use $\mathcal{U}_p^{sla}$ and $\mathcal{U}_l^{sla}$ to represent the price and quality level part of the utility of signed SLA respectively. The utility pair of a signed SLA, referred to as *combined utility pair*, is thus represented as $\mathcal{U}^{sla} = \{\mathcal{U}_p^{sla}, \mathcal{U}_l^{sla}\}$. In Equation.4.8, w_p^i and w_p^o capture the respective influences of asp_i and csp_o on the price part of the resulting utility; while w_l^i and w_l^o capture the respective influences of asp_i and csp_o on the quality level part of the resulting utility. The definition of Equation.4.8 enables us to quantify the degree of collaboration for a signed SLA, and can be used as an offer evaluation criterion, which is referred to as **Evaluation criterion 2** in our research. The agreement is reached if either one of the following conditions is satisfied.

1. On the side of csp_o : If the difference between the asp_i's offer and the csp_o's estimated combined utility pair is within a given range, then csp_o will accept the offer. Formally, we have

$$\sqrt{(\alpha_p^k - \mathcal{U}_p^{estSLA_o})^2 + (\alpha_l^k - \mathcal{U}_l^{estSLA_o})^2} < r_5 \quad \exists 1 \leq k \leq \mathcal{K} \tag{4.9}$$

In Equation.4.9, r_5 reflects the concession degree that the csp_o would like to make in the reached agreement.

2. On the side of asp_i : If the difference between the csp_o's offer and asp_i's estimated combined utility pair is within a given range, then asp_i will accept the offer. Formally, we have

$$\sqrt{(\beta_p^k - \mathcal{U}_p^{estSLA_i})^2 + (\beta_l^k - \mathcal{U}_l^{estSLA_i})^2} < r_6 \quad \exists 1 \leq k \leq \mathcal{K} \tag{4.10}$$

In Equation.4.10, r_6 reflects the degree of concession that the seller would like to make in the reached agreement.

In the above equations, we use the superscripts $\{estSLA_o\}$ and $\{estSLA_i\}$ to represented the estimated combined utility from either the side of csp_o or the side of asp_i. This is due to the reason that a negotiation agent cannot know exactly the negotiation counterpart's optimal utility pair, which is required in Equation.4.8. Therefore, when implementing the **Evaluation criterion 2**, we have to use the estimated counterpart's optimal utility pair.

4.2.3 Estimation-guided Offer Proposition

An accurate estimation of the counterpart's optimal utility pair can help a negotiation agent to capture the counterpart's bargaining position and develop appropriate offer proposition strategies. For instance, according to the **Evaluation criterion 1**, an estimation of the counterpart's optimal utility pair directly points the direction of making concession. On the other hand, with respect to the **Evaluation criterion 2**, an estimated combined utility pair is used as the target point indicating the direction of making concessions. These observations lead us to develop the estimation-guided offer proposition strategy. Figure.4.5 illustrates a snapshot of this type of offer proposition strategy invoked by csp_o when we use **Evaluation criterion 1**.

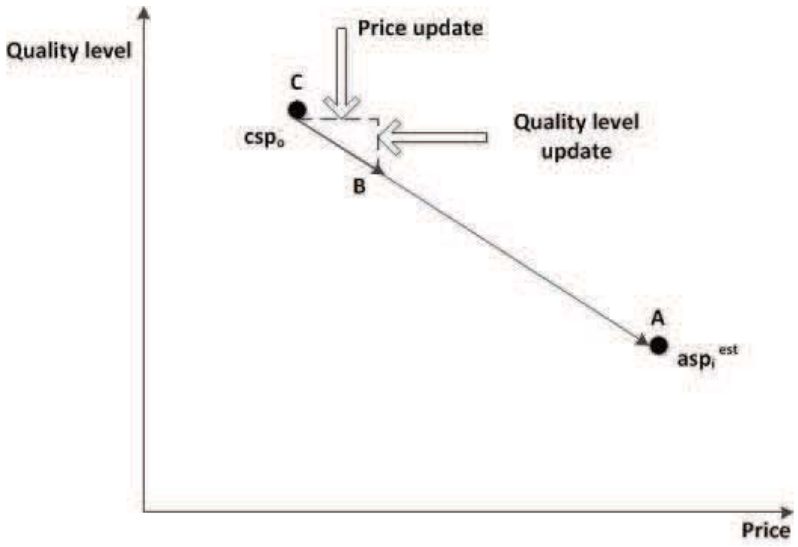

Figure 4.5 Estimation-guided offer proposition

In Figure.4.5, the estimated optimal utility of asp_i is marked as A, we add the superscript est to asp_i to stress the fact that it is not the real optimal utility pair. The real optimal utility pair of csp_o is marked as C. When **Evaluation criterion 1** is used, csp_o will try to approach the asp_i's optimal utility pair along with the direction of $C \rightarrow A$. The distance of CB stands for the degree of the concession. The projection of the concession onto the axis of *price* represents the price update; and the projection onto the axis of *quality level* represents the quality level update. In the bilateral negotiation-based SLA design using Bayes estimator, the estimation of the negotiation counterpart's optimal utility pair will be updated at each iteration based on the counterpart's offer. This is reflected through the ever-changing position of A in the context of the Figure.4.5.

4.2.4 Implementation of Bayes Estimator for the Optimal Utility Pair

Algorithm.4.3 lists the procedures for implementing the Bayes estimator for the optimal utility pair. The objective of Algorithm.4.3 is to compute $\mathbf{E}[\theta^i|\alpha]$. In the two-

dimensional discrete parameter space of $[\theta^i_{p.low}, \theta^i_{p.high}] \otimes [\theta^i_{l.low}, \theta^i_{l.low}]$, we have

$$\begin{aligned} \mathbf{E}[\theta^i|\alpha] &= \mathbf{E}[(\theta^i_p, \theta^i_l)|\alpha] \\ &= \sum_{a=1}^{a=n_1} \sum_{b=1}^{b=m_1} (\theta^{i,a}_p, \theta^{i,b}_l) \mathbf{f}_{posterior}((\theta^{i,a}_p, \theta^{i,b}_l)|\alpha) \end{aligned} \tag{4.11}$$

According to Bayes formula [50],

$$\mathbf{f}_{posterior}((\theta^{i,a}_p, \theta^{i,b}_l)|\alpha) = \frac{\mathbf{f}_{likelihood}(\alpha|\theta^{i,a}_p, \theta^{i,b}_l) * \mathbf{f}_{prior}(\theta^{i,a}_p, \theta^{i,b}_l)}{\mathbf{f}_{marginal}(\alpha)} \tag{4.12}$$

Therefore,

$$\mathbf{E}[\theta^i|\alpha] = \sum_{a=1}^{a=n_1} \sum_{b=1}^{b=m_1} (\theta^{i,a}_p, \theta^{i,b}_l) \frac{\mathbf{f}_{likelihood}(\alpha|\theta^{i,a}_p, \theta^{i,b}_l) * \mathbf{f}_{prior}(\theta^{i,a}_p, \theta^{i,b}_l)}{\mathbf{f}_{marginal}(\alpha)} \tag{4.13}$$

In the above derivations, α represents an offer proposed by asp_i. When we consider the above numerical scheme in the context of a given negotiation iteration, k, α will be replaced with α^k. In order to get $\mathbf{f}_{posterior}((\theta^{i,a}_p, \theta^{i,b}_l)|\alpha)$, we need three different probability distributions: $\mathbf{f}_{prior}(\theta^{i,a}_p, \theta^{i,b}_l)$, $\mathbf{f}_{likelihood}(\alpha|\theta^{i,a}_p, \theta^{i,b}_l)$ and $\mathbf{f}_{prior}(\theta^{i,a}_p, \theta^{i,b}_l)$. The computation procedure of marginal distribution is given in Algorithm.4.3. The computation of these three probability distributions are discussed subsequently.

For simplicity, we just focus on the side of csp_o, i.e., the Bayes estimator of θ^i_p and θ^i_l from the viewpoint of csp_o. The discussion from the other direction is similar.

Prior distribution In accordance with the discretization scheme, the prior distribution is to be considered for each parameter pair $(\theta^{i,a}_p, \theta^{i,b}_l)$. At the very beginning, we do not have any knowledge over the distribution model of the negotiation counterpart's optimal utility pair. Hence, we can just assume it follows uniform distribution over the discrete parameter space. At the k-th negotiation iteration, we treat the counterpart's offer at the (k-1)-th iteration as the knowledge to derive the prior distribution. In other words, we use the posterior distribution of $(\theta^{i,a}_p, \theta^{i,b}_l)$ given α^{k-1} as the prior distribution at the k-th iteration. Therefore, we have

$$\mathbf{f}^k_{prior}(\theta^{i,a}_p, \theta^{i,b}_l) = \mathbf{f}_{posterior}(\theta^{i,a}_p, \theta^{i,b}_l|\alpha^{k-1}) \tag{4.14}$$

Algorithm 4.3 estOUP$_o(\alpha^k, \{\mathbf{f}^k_{prior}(\theta^{i,a}_p, \theta^{i,b}_l) : a = 1...n_1, b = 1...m_1\}, k)$:Bayes estimator for the optimal utility pair of asp_i

1: $\mathbf{f}_{marginal}(\alpha^k) = 0$;
2: **for** $a = 1 : n_1$ **do**
3: **for** $b = 1 : m_1$ **do**
4: $\mathbf{f}_{likelihood}(\alpha^k|\theta^{i,a}_p, \theta^{i,b}_l) = \mathbf{calProb}(\theta^{i,a}_p, \theta^{i,b}_l, \alpha^k, k)$;
5: $\mathbf{f}_{marginal}(\alpha^k) = \mathbf{f}_{marginal}(\alpha^k) + \mathbf{f}_{likelihood}(\alpha^k|\theta^{i,a}_p, \theta^{i,b}_l) * \mathbf{f}^k_{prior}(\theta^{i,a}_p, \theta^{i,b}_l)$;
6: **end for**
7: **end for**
8: $(\widehat{\theta^i_p}, \widehat{\theta^i_l}) = 0$;
9: **for** $a = 1 : n_1$ **do**
10: **for** $b = 1 : m_1$ **do**
11: $\mathbf{f}_{posterior}(\theta^{i,a}_p, \theta^{i,b}_l|\alpha^k) = \frac{\mathbf{f}_{likelihood}(\alpha^k|\theta^{i,a}_p, \theta^{i,b}_l) * \mathbf{f}^k_{prior}(\theta^{i,a}_p, \theta^{i,b}_l)}{\mathbf{f}_{marginal}(\alpha^k)}$;
12: $(\widehat{\theta^i_p}, \widehat{\theta^i_l}) = (\widehat{\theta^i_p}, \widehat{\theta^i_l}) + (\theta^{i,a}_p, \theta^{i,b}_l) * \mathbf{f}_{posterior}(\theta^{i,a}_p, \theta^{i,b}_l|\alpha^k)$;
13: **end for**
14: **end for**
15: **Return** : $[\{\mathbf{f}_{posterior}(\theta^{i,a}_p, \theta^{i,b}_l|\alpha^k) : a = 1...n_1, b = 1...m_1)\}, (\widehat{\theta^i_p}, \widehat{\theta^i_l})]$;

In Equation.4.14, $\alpha^{k-1} = (\alpha^{k-1}_p, \alpha^{k-1}_l)$ stands for the asp_i's offer at the (k-1)-th iteration. $\mathbf{f}_{posterior}(\theta^{i,a}_p, \theta^{i,b}_l|\alpha^{k-1})$ is the posterior distribution, which is to be computed at (k-1)-th iteration. The superscript k of $\mathbf{f}^k_{prior}(\theta^{i,a}_p, \theta^{i,b}_l)$ is used to indicate the fact that it is the prior distribution used at the k-th negotiation iteration. This relationship is reflected in the output parameters at lines 13 and 19 in Algorithm.4.2, and in the input parameter used by Algorithm.4.3.

Likelihood computation The bilateral negotiation is characterized as a process where both negotiation partners will make concessions as the negotiation proceeds. In practice, the starting offer of a negotiation agent tends to enlarge its short term business benefit instead of focusing on long run collaborations and market stabilities. For instance, the $\alpha^1_p \gg \theta^i_p$ and $\alpha^1_l \ll \theta^i_l$. This kind of offer initialization strategy also helps hiding the real optimal utility pair and leave more spaces for bargaining. If the agreement cannot be reached, then the next-step offer will move away from the initialization offer by making concessions in the direction of ap-

proaching the negotiation target. Hence, a theoretically predictable offer at each iteration can be represented as a function of initialization offer and iteration number. We refer this offer as *round-dependent benchmark offer*. For instance, the price part in the round-dependent benchmark offer of asp_i can be computed with the formula like Equation.4.15 when assuming the linear concession strategy.

$$\alpha_p^k = r * \theta_p^i - \rho * (k-1) \tag{4.15}$$

In Eq.4.15, μ stands for the decreasing rate; $r * \theta_p^i$ represents the starting offer and $r > 1$. To capture the unaccounted uncertainties, we can assume that the proposed offer should follow a probabilistic distribution being peak at this round-dependent benchmark offer. A triangular probabilistic distribution or a Gaussisn distribution can be used. The likelihood function, denoted as $\mathbf{f}_{likelihood}(\alpha^k|\theta_p^{i,a}, \theta_l^{i,b})$ stands for the conditional probability of the offer α^k when the optimal utility pair is assumed to be equal to $(\theta_p^{i,a}, \theta_l^{i,b})$. Hence, the variable of θ_p^i in Equation.4.15 should be replaced with $\theta_p^{i,a}$ to compute the corresponding round-dependent benchmark offer.

Marginal distribution The marginal distribution of offer α^k is calculated in Eq.4.16. The operations between line 2 to line 7 in Algorithm.4.3 implement the corresponding computation.

$$\mathbf{f}_{marginal}(\alpha^k) = \sum_{a=1}^{a=n_1} \sum_{b=1}^{b=m_1} \mathbf{f}_{likelihood}(\alpha^k|\theta_p^{i,a}, \theta_l^{i,b}) \mathbf{f}_{prior}^k(\theta_p^{i,a}, \theta_l^{i,b}) \tag{4.16}$$

4.2.5 Experimental Study (1)

In the first experimental setting, we implement $\mathbf{evaOffer}_o$ and $\mathbf{evaOffer}_i$ based on **Evaluation criterion 1**. According to this offer evaluation criterion, the agreement will only be reached either within a small range near (θ_p^i, θ_l^i) or within a small range near (θ_p^o, θ_l^o). One of the main reasons of setting up this evaluation criterion is to test the

ability of estimating the opponent's optimal utility pair. If neither of the negotiation agent is able to estimate the opponent's optimal utility pair accurately, then it will be very hard for the negotiation to settle on an agreement. In this experiment, we consider two scenarios:

Scenario 1 Both csp_o and asp_i employ the Bayes estimator, and hence the SLA design process follows exactly what Algorithm.4.2 guides. The operations of **calOffer**$_o$ and **calOffer**$_i$ used the estimation-guided offer proposition which is illustrated in Figure.4.5.

Scenario 2 None of csp_o and asp_i employs the Bayes estimator, and just uses the central point of the whole domain as the estimation, e.g., $(0.5*(\theta^i_{p.low}+\theta^i_{p.high}), 0.5*(\theta^i_{l.low}+\theta^i_{l.high}))$.

According to **Evaluation criterion 1**, when r_1 increases and r_2 decreases, the criterion of accepting the offer will be relaxed on the side of csp_o. Similarly, when r_3 decreases and r_4 increases, the criterion of accepting the offer will be relaxed on the side of asp_i. We design four different combinations of r_1, r_2, r_3 and r_4, which are identified as Case 1, Case 2, Case 3 and Case 4 respectively in Table.4.1.

Table 4.1 Test on the agreement reachability of the negotiation

	r_1	r_2	r_3	r_4	iterations(scenario 1)	iterations(scenario 2)
Case 1	1.15	0.85	0.85	1.15	86	fail
Case 2	1.18	0.82	0.82	1.18	79	fail
Case 3	1.20	0.80	0.80	1.20	75	fail
Case 4	1.22	0.78	0.78	1.22	70	fail

In Table.4.1, the column of *iterations(scenario 1)* stores the number of iterations required to reach an agreement for the *scenario 1*, and the column of *iterations(scenario 2)* stores the number of iterations required to reach an agreement for the *scenario 2*. If the agreement cannot be reached within a pre-determined deadline, we will mark the corresponding status as *fail*. The result in Table.4.1 shows two facts:

1. The number of iterations required for reaching the agreement decreases as we relax the agreement settling criterion, this matches our expectation.

2. If neither side employs the Bayes estimator, the agreement cannot be reached within the deadline, which is pre-determined as 200 iterations.

The second observation clearly illustrates the advantage of employing the Bayes estimator in estimating the negotiation counterpart's optimal utility pair.

4.2.6 Experimental Study (2)

In the second experimental setting, we implement **evaOffer**$_o$ and **evaOffer**$_i$ based on **Evaluation criterion 2**. In Equation.4.8, the combination of w_p^i, w_p^o, w_l^i and w_l^o reflects how a negotiation agent's business benefit is taken care of in the settled SLA. For instance, a combination of $w_p^i < w_p^o$ and $w_l^i < w_l^o$ indicates that the signed SLA tends to treasure the benefit of csp_o more; or in other words, csp_o has more bargaining power [51]. This is a typical scenario in the competitive service market, where there always have multiple service providers for a signal service. In the study, we consider four different weight combinations:

Combination 1 $w_p^o > w_p^i$ and $w_l^o > w_l^i$, this indicates that in the settling SLA, the price part and the quality level part of the csp_o's benefit are considered more important than the corresponding parts of asp_i. This weight combination is tested in Case 1 and Case 2.

Combination 2 $w_p^o < w_p^i$ and $w_l^o > w_l^i$, this indicates that in the settling SLA, the price part of the asp_i's benefit is considered more important than the corresponding part of csp_o, while the quality level part of the csp_o's benefit is considered more important than the corresponding part of asp_i.This weight combination is tested in Case 3 and Case 4.

Combination 3 $w_p^o > w_p^i$ and $w_l^o < w_l^i$, this indicates that in the settling SLA, the price part of the csp_o's benefit is considered more important than the corresponding part of asp_i, while the quality level part of the asp_i's benefit is considered more important than the corresponding part of csp_o. This weight combination is tested in Case 5 and Case 6.

Combination 4 $w_p^o < w_p^i$ and $w_l^o < w_l^i$, the price part and the quality level part of the asp_i's benefit are considered more important than the corresponding parts of csp_o. This weight combination is tested in Case 7 and Case 8.

Table 4.2 Test on the ability of achieving the target utility

	$\{w_p^i, w_p^o\}$	$\{w_l^i, w_l^o\}$	$\{\mathcal{U}_p^{slaT}, \mathcal{U}_l^{slaT}\}$	$\{\mathcal{U}_p^{sla1}, \mathcal{U}_l^{sla1}\}$	$\{\mathcal{U}_p^{sla2}, \mathcal{U}_l^{sla2}\}$
Case 1	$\{0.30, 0.70\}$	$\{0.40, 0.60\}$	$\{870, 71\}$	$\{864.641, 71.661\}$	$\{811.612, 76.155\}$
Case 2	$\{0.20, 0.80\}$	$\{0.30, 0.70\}$	$\{830, 74.5\}$	$\{830.939, 74.017\}$	$\{801.414, 77.453\}$
Case 3	$\{0.70, 0.30\}$	$\{0.40, 0.60\}$	$\{1030, 71\}$	$\{1035.271, 68.596\}$	$\{1076.497, 68.280\}$
Case 4	$\{0.80, 0.20\}$	$\{0.30, 0.70\}$	$\{1070, 74.5\}$	$\{1070.354, 71.541\}$	$\{1082.343, 73.884\}$
Case 5	$\{0.20, 0.80\}$	$\{0.70, 0.30\}$	$\{830, 60.5\}$	$\{825.402, 60.494\}$	$\{814.466, 61.215\}$
Case 6	$\{0.30, 0.70\}$	$\{0.60, 0.40\}$	$\{870, 64\}$	$\{867.560, 63.953\}$	$\{808.404, 67.632\}$
Case 7	$\{0.80, 0.20\}$	$\{0.70, 0.30\}$	$\{1070, 60.5\}$	$\{1096.221, 55.753\}$	$\{954.157, 70.005\}$
Case 8	$\{0.70, 0.30\}$	$\{0.60, 0.40\}$	$\{1030, 64\}$	$\{1043.559, 61.123\}$	$\{982.759, 67.939\}$

In Table.4.2, the second and the third column report the weight combinations. Besides these parameter settings, we also report the following set of results:

1. The fourth column reports the target combined utility pair for the settled SLA, $\{\mathcal{U}_p^{slaT}, \mathcal{U}_l^{slaT}\}$. We submitted the real optimal utility pairs of both csp_o and asp_i into Equation.4.8. Since an agent's optimal utility pair is the hidden knowledge to its counterpart, this target pair only exists theoretically. We use it as the benchmark in the experimental study.

2. The fifth column reports the result of the SLA designed based on Bayes estimator. Here, we assume both csp_o and asp_i employ the Bayes estimator. The achieved utility pair is represented as $\{\mathcal{U}_p^{sla1}, \mathcal{U}_l^{sla1}\}$.

3. The sixth column reports the result of the SLA designed without using Bayes estimator. The achieved utility pair is represented as $\{\mathcal{U}_p^{sla2}, \mathcal{U}_l^{sla2}\}$.

In terms of the difference between the reached utility and the target utility, the results in Table.4.2 clearly demonstrate that the SLA design employing Bayes estimator outperforms the one without using Bayes estimator in every weight combination.

4.2.7 Related Work and Discussion

Negotiation is an important mechanism to establish business relationships among different market players. Several existing works have been developed focusing on various facets of the negotiation process design. Li et al.[51] investigate the effect of the outside options on the reservation price and bargaining power. Zeng et al.[52] propose to use Bayesian analysis to assist the negotiation process by estimating the negotiation counterpart's reservation price. Sim et al.[53] further developed the Bayesian learning-based negotiation process and took into account the negotiation deadline as an estimation variable in order to improve the negotiation performance. Our work can be considered as a further development on the negotiation process design using Bayesian analysis. Moreover, we extend the existing results and develop the corresponding numerical scheme in the context of multi-attributes SLA design for the competitive service market. Our contributions are listed in the following.

- In the context of competitive service market, we clarify the specific SLA design scenario to which the bilateral negotiation process can be applied.

- In Equation.4.4, we derive the Bayes estimator in the context of the bilateral negotiation based on the statistical inference theory. The mathematical meaning of the derived formula clearly demonstrates how does the counterpart's offer determine the estimation of its optimal utility pair. This type of rigorous derivation is not given in [52] and [53].

- Our research focuses on the multi-attributes SLA design which requires us to consider the negotiation process in a two-dimensional space. We systematically discuss how to design the numerical scheme for the corresponding Bayes estimator.
- With the help of the information provided by the Bayes estimator, we propose the estimation-guided offer proposition strategy. Its flexibility is demonstrated in different offer evaluation criteria.

4.3 Bidding Protocol for Target-Oriented SLA Design

The bilateral negotiation process does not fit well to the target-oriented SLA design because it does not stipulate any specific metric values beforehand. In this section, we develop a bidding protocol for the target-oriented SLA design which is composed of the following procedures.

- Given an atomic service AS_t, composite service provider csp_o notifies the providers belonging to $\mathcal{CG}_o^t$ of its requirements on one or more specific QoS metrics.
- Each related atomic service provider estimates the operation cost with respect to the received QoS requirements, and proposes a bid rate for signing SLA in accordance with the received requirements.
- csp_o selects the atomic service provider which bids with the smallest rate.

From the standpoint of an atomic service provider which joins the bid, reducing the proposed bid rate can increase the probability of being selected by the composite service provider. Nevertheless, charging the buyer with a small rate results in the reduced net profit for itself. Therefore, achieving the balance between the winning probability and the net profit is a major concern in the bidding protocol design.

Theoretically, if $asp_i \in \mathcal{CG}_o^t$ knows exactly what other competitors will propose, then the bid rate of ensuring its winning will be MIN($\xi_j \ : \forall \ asp_j \in \mathcal{CG}_o^t$ and $j \neq i$) $- \delta$,

where ξ_j denotes the bid rate of asp_j and δ is a very small positive value. Nevertheless, a market player tends to hide the sensitive information from its competitors, especially in a competitive bidding process. Consequently, it is infeasible to compute the best bid rate explicitly, which depends on knowing exactly the bid rate of others. In this research, we cast the bidding protocol for the target-oriented SLA design in the Bayesian game theoretic framework and investigate its analysis process. This approach allows the existence of incomplete information and captures the resulting randomness of the competition [54].

A Bayesian game for the bidding protocol between csp_o and a set of its controlled atomic service providers is described by a quadruple $\mathcal{BG} = \{\mathcal{N}, \xi, \Theta, \mathcal{U}\}$.

- $\mathcal{N} = asp_1 \times \cdots \times asp_m$ is the agent space, where n is the number of participating game players, and asp_i denotes the i-th game player which is an atomic service provider participating in the bidding game.
- $\xi = \xi_1 \times \cdots \times \xi_m$ is the action space, where ξ_i is the action of game player asp_i.
- $\theta = \theta_1 \times \cdots \times \theta_m$ is the type space, where θ_i is the set of type values of asp_i.
- $\mathcal{U} = \mathcal{U}_1 \times \cdots \times \mathcal{U}_m$ is the utility space, where $\mathcal{U}_i$ is the utility function of game player asp_i;

In the above, $m \leq= \|\mathcal{CG}_o^t\|$ since not all of the atomic service providers is necessary to join the bidding game. The action of each game player is its bid rate. The type value of a game player reflects its private valuation over delivering the service. In our case, the real type value of asp_i is its operation cost needed to satisfy the target QoS metric values. In the competitive service market, this exact type value is hidden from other agents, $asp_j : j \neq i$. However, the accessibility to the public market information resources and other bidders' behaviors enables an agent to derive a probabilistic distribution of either the type values or the bid values of other agents. The existences of these

different probabilistic models lead to various approaches of bidding protocols, which will be discussed later in this section.

In the competitive service market, each market player is assumed to behave as a rational agent, whose action is motivated by maximizing its utility. In the bidding game, the utility of asp_i is computed in Equation.4.17.

$$\mathcal{U}(asp_i) = \mathbf{I}_{i=\text{argmin}(\xi_1,\ldots,\xi_m)} \times (\xi_i - \theta_i) \tag{4.17}$$

where

$$\mathbf{I}_{i=\text{argmin}(\xi_1,\ldots,\xi_m)} = \begin{cases} 1 & \xi_i < \xi_j \ \ \forall \ \ j \neq i \\ 0 & \text{otherwise} \end{cases}$$

Equation.4.17 illustrates that, if ξ_i, the bid of asp_i, is the minimum among all of the game players, then asp_i will be selected as the winner, and thus its payoff is equivalent to $\xi_i - \theta_i$; otherwise, its payoff equals to 0.

In the above, we discuss the tradeoff between the selection probability and the resulting profit. The following analysis will quantify this tradeoff probabilistically. If asp_i bids with rate ξ_i, then the winning probability will be determined by the possibility that none of other competitors bids with a rate being smaller than asp_i. According to the bidding game setting, the probabilistic model for the action space is a common knowledge to the game players. Here, we assume that the real bid rate of every agent follows the uniform distribution on a shared range, $[\omega_1, \omega_2]$. Thus, the probability of bidding with ξ_i is $\frac{1}{\omega_2 - \omega_1}$; the probability of bidding with value being bigger than ξ_i is $\int_{\xi_i}^{\omega_2} \frac{1}{\omega_2 - \omega_1} dy$. Accordingly, the expected utility is computed as

$$\begin{aligned} \mathbf{E}[\mathcal{U}(asp_i)] &= \mathbf{E}[\mathbf{I}_{i=\text{argmin}(\xi_1,\ldots,\xi_m)} \times (\xi_i - \theta_i)] \\ &= \mathbf{Pr}(\xi_i < \xi_j \ \forall \ j \neq i) \times (\xi_i - \theta_i) \end{aligned} \tag{4.18}$$

since

$$\begin{aligned} \mathbf{Pr}(\xi_i < \xi_j \quad \forall\, j \neq i) &= (\int_{\xi_i}^{\omega_2} \frac{1}{\omega_2 - \omega_1} dy)^{m-1} \\ &= \frac{(\omega_2 - \xi_i)^{m-1}}{(\omega_2 - \omega_1)^{m-1}} \end{aligned} \tag{4.19}$$

so

$$\mathbf{E}[\mathcal{U}(asp_i)] = \frac{(\omega_2 - \xi_i)^{m-1}}{(\omega_2 - \omega_1)^{m-1}} \times (\xi_i - \theta_i) \tag{4.20}$$

The rate ξ_i that can maximize $\mathbf{E}[\mathcal{U}(asp_i)]$ is obtained by solving

$$\frac{d\ \mathbf{E}[\mathcal{U}(N_i)]}{d\ \xi_i} = 0 \tag{4.21}$$

After substituting the result of Equation. 4.20 into Equation. 4.21, we get

$$\xi_i = \frac{\omega_2 + (m-1)\theta_i}{m} \tag{4.22}$$

Equation.4.22 explicitly stipulates the rate that a game player would like to bid in order to maximize the expected payoff when only incomplete information is ready. Consider a test case, which is:

- The operation cost of every participating agent is sampled from a uniform distribution. In Table 4.3, the generated values are listed in the row indexed by "θ_i(uniform)". Moreover, from the viewpoint of a participating atomic service provider, the bid rate of any competitors is assumed to follow a uniform distribution on a known range. Therefore, we can use formula given in Equation.4.22 to compute the bid rate.

Table 4.3 Experimental study of bid rate

The first scenario					
	asp_1	asp_2	asp_3	asp_4	asp_5
θ_i(uniform)	111.5	129.2	142.2	123.8	67.6
ξ_i of scenario 1	119.2	133.4	143.7	129.1	84.1

The results in Table.4.3 shows that the proposed bid rate of each atomic service provider keeps the trend of its type value. asp_5 will be chosen as the winner due to its having the lowest bid rate, which is controlled by its type value, θ_5. However, there are several issues affecting the applicability of this approach. The analytical formula only exists for some simple probabilistic models, like uniform distribution. Secondly, the net profit of the winner still has a lot of space to get improved. With respect to this particular test case, the type value of asp_1 is 111.5. Theoretically, asp_5 can increase its bid to this value and can still win the auction. However, this relies on a more accurate estimation of other competitor's bid. In the following, we are going to investigate these issues.

4.3.1 Monte Carlo-Based Bidding Protocol Design

The closed-form bidding strategy discussed in the above only works for the scenario where every competitor's bid rate is assumed to follow simple probabilistic models, e.g., uniform distribution. With respect to the more complicated probabilistic models, such as normal distribution and exponential distribution, the analytical formula is impossible or at least very hard to get. Here, we investigate the application of Monte Carlo method in the bidding protocol design [55].

The difficulty in deriving the analytical bidding strategy is that the integration in Equation.4.19 will become unsolvable once the adopted probabilistic model is very complicated. To solve this issue, Monte Carlo method is to sample from the underlying probabilistic distribution, and transforms the original computation in a continuous space into the computation in a discrete space. Algorithm.4.4 is a Monte Carlo-based procedure for calculating the bid rate

In Algorithm.4.4, ξ_{-i} represents the set used to store the bid rate of the atomic service providers which are asp_i's competitors; $\mathcal{K}$ is the number of Monte Carlo iterations. We use $\mathcal{B}_{-i}$ to denote the set of asp_i's competitors and use $\mathbf{f}_{asp_j}$ to denote the probability distribution of the bid rate for asp_j. From the viewpoint of asp_i employing **mcBid**, it

Algorithm 4.4 mcBid($\mathcal{K}, \{\mathbf{f}_{asp_j} : asp_j \in \mathcal{B}_{-i}\}$):Monte Carlo-based bidding strategy

1: $\xi_{-i} = \emptyset$;
2: **for** $k = 1 : \mathcal{K}$ **do**
3: **for** $asp_j \in \mathcal{B}_{-i}$ **do**
4: Sample a bid value b_j from the probability distribution, $\mathbf{f}_{asp_j}$;
5: $\xi_{-i} = \xi_{-i} \cup b_j$;
6: **end for**
7: $\xi_i^k =$ **estCalBid**(ξ_{-i}, θ_i);
8: **end for**
9: Compute the bid rate of asp_i as $\xi_i = \frac{\sum_{k=1}^{k=\mathcal{K}} \xi_i^k}{\mathcal{K}}$;

has to know the probability distribution of the bid rate of its competitors beforehand. This is the same prerequisite as the one to derive the analytical formula. But it does not need analytically solving an equation involving the assumed probabilistic distributions. The operation of **estCalBid** is to calculate the bid of asp_i. This operation is listed in Procedure.4.1, whose input parameters are the estimated bid rates of its competitors, which are collected through sampling from the known probability distributions in Algorithm.4.4.

Procedure 4.1 estCalBid(ξ_{-i}, θ_i): Procedure of calculating asp_i's bid based on the bid rates of competitors

1: **if** $\theta_i = \text{MIN}(\xi_{-i} \cup \theta_i)$ **then**
2: $\xi_i = \text{MIN}(\xi_{-i}) - \delta$;
3: **else**
4: $\xi_i = \theta_i + \sigma$;
5: **end if**

Procedure.4.1 calculates asp_i's bid based on the sampled bid rates of its competitors. If its type value, θ_i is lower than any of these bid rates, the bidding strategy of ensuring its winning is to approach the lowest bid rate of its competitors.This is what line 2 in Procedure.4.1 represents. δ is a small numerical value introduced in the implementation, and its introduction is to remove the possibility of having two equivalent lowest bid rates. If asp_i finds that its type value is higher than the lowest bid rate of its competitors, then it cannot win the auction due to its larger type value. With respect to this scenario, it

just add a finite increment,σ, to its type value as an approach to hide its real type value and deceive its competitors.

4.3.1.1 Experimental Study: One-Dimensional Normal Distribution

We will investigate the adoption of Monte Carlo-based bidding strategy in this section and the following section. Instead of assuming simple uniform probability distribution, we will consider more general ones: one is the one-dimensional Normal distribution and the other one is the bivariate normal distribution.

In this experiment, we assume that $\xi_j \sim Normal(\mu, \sigma)$. The corresponding probability density function is given in Equation.4.23.

$$\mathbf{f}(\xi_j) = \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{(\xi_j - \mu)^2}{2\sigma^2}) \tag{4.23}$$

In Equation.4.23, μ is the mean of the bid rate and σ^2 is the variance. This probability information of asp_j is held by asp_i.

The experiment is setup as follows:

1. The auction has three participating atomic service providers, i.e., asp_1, asp_2 and asp_3.

2. The probability distribution of the bid rate is assumed to be the same for the three atomic service providers. If the mean bid rates vary with different atomic service providers, then the one with the smallest mean value tends to win the auction most of the time no matter whether the Monte Carlo-based bidding strategy is employed or not. In order to validate the effectiveness of the Monte Carlo-based bidding strategy, we try to filter out any factors that may affect the auction result.

3. Two tests cases are designed for the comparison. In **test case 1**, none of the participating atomic service providers employs the Monte Carlo-based bidding strategy. The only information that can help an agent to propose its bid is the mean bid rate

of the competitors. In **test case 2**, asp_1 employs the Monte Carlo-based bidding strategy, i.e., Algorithm.4.4, while asp_2 and asp_3 only rely on the mean value in the bidding process.

The simulation procedure in **test case 1** is listed in Procedure.4.2. Since none of the atomic service providers adopts the Monte Carlo-based bidding strategy, the parameters used in **estCalBid** are the mean bid rate of competitors. **findWinner** is a generic function of determining the winner based on the proposed bids. bW represent the auction winner and sP represents the selling price. In this research, the agent proposing the lowest bid rate is chosen as the winner and its bid is used as the selling price.

Procedure 4.2 **testcase1Simulation**: Simulation procedure of **test case 1**

1: **for** $i = 1 : 3$ **do**
2: $\quad \xi_i = \textbf{estCalBid}(\{\mu_j : asp_j \in \mathcal{B}_{-i}\}, \theta_i)$;
3: **end for**
4: $(bW, sP) = \textbf{findWinner}(\xi_1, \xi_2, \xi_3)$;

The simulation procedure in **test case 2** is listed in Procedure.4.3 which differs with Proc.4.2 in the bid computation of asp_1. It is assumed to adopt the Monte Carlo-based bidding strategy, i.e., **mcBid**. The experimental results are reported in Figure.4.6.

Procedure 4.3 **testcase2Simulation**: Simulation procedure of **test case 2**

1: $\xi_1 = \textbf{mcBid}(\mathcal{K}, \mathbf{f}_{asp_2}, \mathbf{f}_{asp_3})$;
2: **for** $i = 2 : 3$ **do**
3: $\quad \xi_i = \textbf{estCalBid}(\{\mu_j : asp_j \in \mathcal{B}_{-i}\}, \theta_i)$;
4: **end for**
5: $(bW, sP) = \textbf{findWinner}(\xi_1, \xi_2, \xi_3)$;

In the experiment, we consider 50 independent bid cases. Among these 50 bid cases, we record every bid case where asp_1 is chosen as winner. Let $\gamma_1 = sP - \theta_1$ denote the net profit of asp_1 in **test case 1**; similarly, γ_2 denotes the net profit of asp_1 in **test case 2**. Hence, $\gamma_2 - \gamma_1$ denotes the improved profit when adopting the Monte Carlo-based bidding strategy. These improved profits are marked with "square" point in the

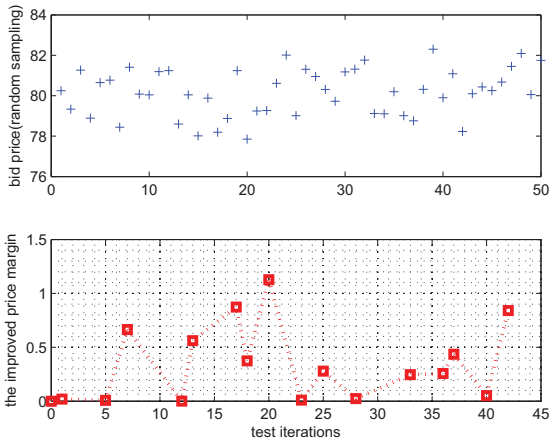

Figure 4.6 Experimental result for normal distribution($var = 1$)

lower part of Figure.4.6. The upper part of Figure.4.6 records a collection of bid rates following the assumed distribution, i.e., $Normal(80, 1)$.

Secondly, we consider a different probability model where the bid rate is assumed to follow $Normal(80, 5)$ for every atomic service provider. This probability distribution is to test whether the Monte Carlo-based bidding strategy will still work for a large variance model. The test result is shown in Figure.4.7.

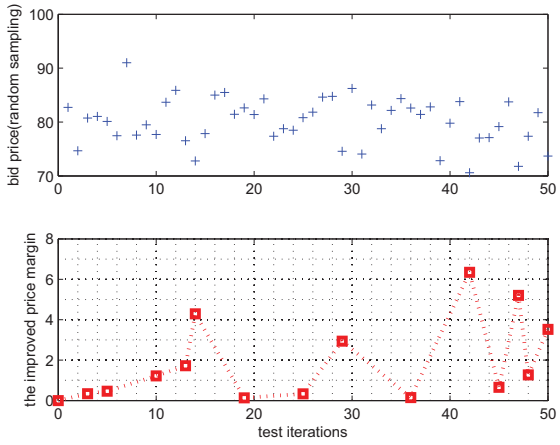

Figure 4.7 Experimental result for normal distribution($var = 5$)

For every bid case where asp_1 wins the auction, no negative value is observed either in Figure.4.6 or in Figure.4.7. This observation demonstrates that the employment of Monte Carlo-based bidding strategy design can improve the profit margin of asp_1.

4.3.1.2 Experimental Study:Two-dimensional Bidding Strategy Design

In this section, we investigate the application of Monte Carlo-based bidding strategy design in the two-dimensional space, where we consider both the price and the quality level. The proposed bid is thus represented as a pair (ξ_p^i, ξ_l^i), where ξ_p^i represents the bid rate and ξ_l^i represents the bid quality level. This study aims to support the multi-attributes target-oriented SLA design.

In the one-dimensional auction for the target-oriented SLA design, the utility function of the composite service provider fully depends on the selling price of the auction winner. In the two-dimensional auction, the utility function of the composite service provider depends on both the selling price and the quality level, and can be represented as in Equation.4.24.

$$\mathcal{U}_{csp}(\mathcal{P}, \mathcal{L}) = -w_p \times \frac{\mathcal{P}}{\mathscr{P}} + w_l \times \frac{\mathcal{L}}{\mathscr{L}} \tag{4.24}$$

In Equation.4.24, w_p and w_l are the weight factors for the price and quality level respectively. The minus sign before w_p indicates that a higher selling price leads to a smaller utility for the composite service provider. The $\mathscr{P}$ and $\mathscr{L}$ on the denominator is to normalize price $\mathcal{P}$ and quality level $\mathcal{L}$ since their numerical values may have different scales. In the two-dimensional auction for the target-oriented SLA design, the utility represented in Equation.4.24 will be used as the criterion to guide the atomic service provider to calculate its bid rate.

Comparing to the bid calculation procedure in the one-dimensional auction, the bid calculation procedure in the two-dimensional auction is more complicated due to its inherent multiple choices resulting from the utility formulation. Figure.4.8 illustrates

how the bid computation scheme works in the two-dimensional auction.

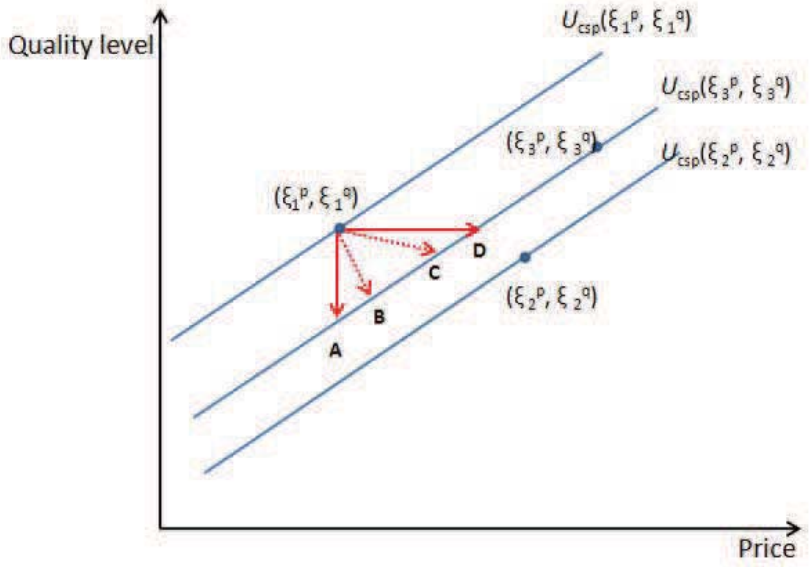

Figure 4.8 Update the bid in the two-dimensional auction

It is assumed to have three atomic service providers. asp_1 bids with (ξ_p^1, ξ_l^1); asp_2 bids with (ξ_p^2, ξ_l^2) and asp_3 bids with (ξ_p^3, ξ_l^3). The three points corresponding to the bid pairs are marked in the figure. For each point, we also plot the line sharing the same utility with it. For instance, the equation of line where (ξ_p^3, ξ_l^3) lies is formulated as

$$-w_p \times \frac{\xi_p^x}{\mathscr{P}} + W_q \times \frac{\xi_l^x}{\mathscr{L}} = -W_p \times \frac{\xi_p^3}{\mathscr{P}} + W_q \times \frac{\xi_l^3}{\mathscr{L}}. \tag{4.25}$$

Every point locating on this line, e.g., A, B, C and D share the same utility as the one resulting from the bid (ξ_p^3, ξ_l^3). After substituting these bids into Equation.4.24, we have $\mathcal{U}_{csp}(\xi_p^1, \xi_l^1) > \mathcal{U}_{csp}(\xi_p^3, \xi_l^3) > \mathcal{U}_{csp}(\xi_p^2, \xi_l^2)$. This relationship is also reflected by three parallel lines in Figure.4.8. From the viewpoint of asp_1, bidding with (ξ_p^1, ξ_l^1) ensures its being selected as the winner. However, its most preferable result is to guarantee its winning position while maximizing its own utility by increasing the selling price, downgrading the quality level or both. The example showing in Figure.4.8 gives three choices:

- A, having the same price with point (ξ_p^1, ξ_l^1), represents the target update point if asp_1 plans to downgrade the quality level only.
- D, having the same quality level with point (ξ_p^1, ξ_l^1), represents the target update point if asp_1 plans to increase the selling price only.
- B, C and other points locating between A and D, represent the target update points if asp_i plans to increase the selling price and downgrade the quality level at the same time.

In the experimental study, we assume that (ξ_p^i, ξ_l^i) follows the bivariate normal distribution, which is $\mathcal{N}(\mu, \Sigma)$ [47].

$$\mu = [\text{mean}(\xi_p^i), \text{mean}(\xi_l^i)] \qquad \Sigma = \begin{bmatrix} \text{Var}(\xi_p^i) & \text{Cov}(\xi_p^i, \xi_l^i) \\ \text{Cov}(\xi_p^i, \xi_l^i) & \text{Var}(\xi_l^i) \end{bmatrix} \tag{4.26}$$

In Equation.4.26, $\text{mean}(x)$ represents the mean value of x, $\text{Var}(x)$ is the variance of x and $\text{Cov}(x, y)$ is the covariance of x and y. In the study, the concrete parameters are setup as:

$$\begin{aligned} [\text{mean}(\xi_p^i), \text{mean}(\xi_l^i)] &= [80, 65] \\ \begin{bmatrix} \text{Var}(\xi_p^i) & \text{Corr}(\xi_p^i, \xi_l^i) \\ \text{Corr}(\xi_p^i, \xi_l^i) & \text{Var}(\xi_l^i) \end{bmatrix} &= \begin{bmatrix} 3 & 0.3 \\ 0.3 & 3 \end{bmatrix} \quad \forall\ i = 1, 2, 3 \end{aligned} \tag{4.27}$$

Figure.4.9 presents the result for this test case. Similar to the above study on the one-dimension auction, the upper part of the figure is the diagram of the data, and the lower part marks the improved net profit with the associated bid case. The benchmark case is that every atomic service provider computes the bid based on the trivial estimation, i.e., the mean values of the competitors' bid pairs. The investigated tested case is that asp_1 adopts the Monte Carlo-based bidding strategy while asp_2 and asp_3 do not. The results show that nine bid cases see clear improvement on the net profit for the asp_1

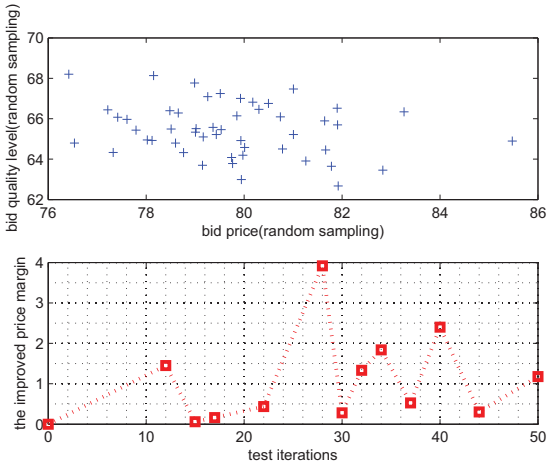

Figure 4.9 Experimental result for bivariate normal distribution

when adopting the Monte Carlo-based bidding strategy design. Here, we use the price-dominant optimal bid computation, e.g., $(\xi_p^1, \xi_l^1) \rightarrow D$ in Figure.4.8.

4.3.2 Target-Oriented SLA Design in Sequential Auction

With respect to the target-oriented SLA design, the similar auction process can happen repeatedly involving different service requests arriving sequentially along with the same set of atomic service providers. This scenario can be framed as a specific SLA design setting: target-oriented SLA design using sequential auction.

In the previous section, we discuss the auction-based target-oriented SLA design, which essentially is a one-shot Bayesian game. In practice, just as in the bilateral negotiation where an agent can learn the negotiation counterpart's hidden information from its proposal, a bidding agent can accumulate auction experiences by learning the competitors' bidding behaviors and use this experience to guide the bidding process in the future auction. The existence of this type of learning capability cannot help the one-shot auction, but will help the decision making process in the sequential auction-based SLA design. In this section, we are going to investigate the sequential auction using

Bayes estimator, which is an extension of our previous research in bilateral negotiation.

4.3.2.1 Adaptive Bidding Strategy for the Target-Oriented SLA Design in Sequential Auction

A common practice for a bidder attending the sequential auction is to start from an initialization bid, which usually is much higher than its type value, and decrease the bid in terms of a pre-selected rate if it cannot win the current auction [56]. This common practice can be further refined and allow the bidder to adapt the bidding strategy in a more systematic way.

Suppose the bidder does not hold any advanced learning capabilities, like Bayes estimator, the only information it can exploit is the result of the previous auction. Let pair $\{\mathcal{W}(n), \mathcal{P}(n)\}$ denote the winner and its bid rate in the n-th auction respectively. The winner's bid rate is also referred to as closing price hereafter. There have three different scenarios:

1. Assume asp_i wins the n-th auction, then with the same auction environment, asp_i can increase its bid rate to enlarge the net profit in the (n+1)-th auction. The reason behind this inference is that the bid rate proposed in the n-th auction by asp_i already ensures winning the auction, and there usually exists a gap between this winning bid and the next smallest bid. asp_i can exploit this fact by assuming that this gap will not totally disappear even the competitor may decrease the bid rate in the (n+1)-th auction.

2. Assume asp_i does not win the n-th auction but the closing price is higher than the type value of asp_i, this fact indicates that there still exists possibility to allow asp_i to win the future auctions. However, it should decrease the bid rate from the current closing price.

3. Assume asp_i does not win the n-th auction and the closing price is lower than the type value of asp_i, this fact theoretically removes any potential winnings for asp_i because at least one of its competitors already manages to bid with a rate being lower than its type value. With respect to this scenario, we just let it decrease the bid with a pre-selected rate and to exploit any possibility of winning the future auctions. The underlying motivation of this design is that a winning bidder may increase its bid rate in the next auction. Certainly, this decreasement will use the type value as the bottom line.

This analysis is formalized in Procedure.4.4.

Procedure 4.4 **saCalBid**$(\mathcal{W}(n-1), \mathcal{P}(n-1), \xi_i(n-1), \theta_i)$:Adaptive procedure of calculating $\mathcal{B}_i$'s bid for the n-th auction

1: **if** $asp_i = \mathcal{W}(n-1)$ **then**
2: $\quad \xi_i(n) = \mathcal{P}(n-1) + \mathcal{D}_i^1$;
3: **else if** $\mathcal{P}(n) > \theta_i$ **then**
4: $\quad \xi_i(n) = \mathcal{P}(n-1) - \mathcal{D}_i^2$;
5: **else**
6: $\quad \xi_i(n) = \xi_i(n-1) - \mathcal{D}_i^3$;
7: **end if**
8: **Return** $\xi_i(n)$;

In Procedure.4.4, $\mathcal{W}(n-1)$ and $\mathcal{P}(n-1)$ are the auction winner and selling price in the (n-1)-th auction respectively. The calculated $\xi_i(n)$ will be returned and used by asp_i as the bid in the n-th auction. $\mathcal{D}_i^1$, $\mathcal{D}_i^2$ and $\mathcal{D}_i^3$ correspond to the three different bid changement discussed in the above respectively.

4.3.2.2 Bayes Estimator-based Bidding Strategy for the Target-oriented SLA Design in Sequential Auction

In this section, we study the bidding strategy design for the target-oriented SLA design in sequential auction. The algorithm is listed in Algorithm.4.5.

Algorithm 4.5 bayesBid$(\xi_j(n-1), n, \mathcal{W}(n-2), \mathcal{P}(n-2))$:Bayes estimator-based bidding strategy for the n-th auction round

for $asp_j \in \mathcal{B}_{-i}$ **do**
  for $l = 1 : n_j$ **do**
    if $n = 3$ **then**
      $\mathbf{f}_{prior}(\theta_j^l) = \frac{1}{n_j}$;
    else
      $\mathbf{f}_{prior}(\theta_j^l) = \mathbf{f}_{posterior}(\theta_j^l|\xi_j(n-1))$;
    end if
    $\mathbf{f}_{likelihood}(\xi_j(n-1)|\theta_j^l) = \mathbf{compLikelihood}(\xi_j(n-1), \xi_j(n-2), \mathcal{W}(n-2), \mathcal{P}(n-2), \theta_j^l)$;
  end for
  $\mathbf{f}_{marginal}(\xi_j(n-1)) = 0$;
  for $l = 1 : n_j$ **do**
    $\mathbf{f}_{marginal}(\xi_j(n-1)) = \mathbf{f}_{marginal}(\xi_j(n-1)) + \mathbf{f}_{likelihood}(\xi_j(n-1)|\theta_j^l) \times \mathbf{f}_{prior}(\theta_j^l)$;
    $\mathbf{f}_{posterior}(\theta_j^l|\xi_j(n-1)) = \frac{\mathbf{f}_{likelihood}(\xi_j(n-1)|\theta_j^l) \times \mathbf{f}_{prior}(\theta_j^l)}{\mathbf{f}_{marginal}(\xi_j(n-1))}$;
  end for
  $\tilde{\theta}_j = 0$;
  for $l = 1 : n_j$ **do**
    $\tilde{\theta}_j = \tilde{\theta}_j + \mathbf{f}_{posterior}(\theta_j^l|\xi_j(n-1)) \times \theta_j^l$;
  end for
end for
$\xi_i(n) = \mathbf{typeCalBid}(\{\tilde{\theta}_j \ : \ \forall asp_j \in \mathcal{B}_{-i}\}, \theta_i)$;
Return: $\xi_i(n)$;

Algorithm.4.5 is assumed to be invoked by asp_i to calculate the bid rate to be proposed in the n-th auction. It needs the result of the (n-2)-th auction and the bid rate of the (n-1)-th auction. On the other hand, Procedure.4.4 only needs the result of the (n-1)-th auction to compute the bid for the n-th auction. Hence, Algorithm.4.5 starts to work only after $n > 2$. Similar to the Bayesian analysis in Algorithm.4.2, a collection of candidate values for the real θ_j is constructed by the discretization operation over the range of θ_j, which is $[\theta_j^{low}, \theta_j^{high}]$. θ_j represents the type value of asp_j. We denote the size of this collection as n_j.

$$(\theta_j^l : l = 1 : n_j) = \textbf{discreteSpace}([\theta_j^{low}, \theta_j^{high}]) \tag{4.28}$$

In Algorithm.4.5, **typeCalBid** computes the bid based on the estimated type values of its competitors, and is listed in Procedure.4.5. It differs with Procedure.4.1 in using type values instead of using the estimated bid rates.

Procedure 4.5 **typeCalBid**(θ_{-i}, θ_i): Procedure of calculating asp_i's bid based on the type values of competitors

```
1: if θ_i = MIN(θ_{-i} ∪ θ_i) then
2:     ξ_i = MIN(θ_{-i}) − δ;
3: else
4:     ξ_i = θ_i + σ;
5: end if
```

The segment from "Line 2" to "Line 18" is the core of the Bayes estimator-based bidding strategy design, which computes the posterior distribution of θ_j^l given $\xi_j(n-1)$, the observed bid rate of asp_j in the (n-1)-th auction. The computation formula of the posterior distribution is given in Equation.4.29, which requires the likelihood distribution and prior distribution as the inputs.

$$\mathbf{f}_{post}(\theta_j^l|\xi_j(n-1)) = \frac{\mathbf{f}_{likelihood}(\xi_j(n-1)|\theta_j^l) \times \mathbf{f}_{prior}(\theta_j^l)}{\sum_{l=1}^{l=n_j} \mathbf{f}_{likelihood}(\xi_j(n-1)|\theta_j^l) \times \mathbf{f}_{prior}(\theta_j^l)} \tag{4.29}$$

In Algorithm.4.5, the function of **compLikelihood** is used to compute the likelihood

of $\xi_j(n-1)$ given the type value being equal to θ_j^l. This computation is divided into two steps:

1. First, we compute the projected bid of asp_j in the (n-1)-th auction when its type value is assumed to be θ_j^l. This projected bid rate, denoted as $\widehat{\xi_j(n-1)}$, can be obtained by calling the computational procedure of **saCalBid**$(\mathcal{W}(n-2), \mathcal{P}(n-2), \xi_j(n-2), \theta_j^l)$.

2. Secondly, we use a triangular distribution, with peak at the position of $\widehat{\xi_j(n-1)}$, to compute the likelihood of $\xi_j(n-1)$. The rhythm underlying this consideration is that if θ_j^l is close to the real type value, then the real bid value, $\xi_j(n-1)$ will also be close to the projected bid, $\widehat{\xi_j(n-1)}$.

The prior distribution is computed in an updated scheme which is similar to the one used in computing the Bayes estimator for the bilateral negotiation. The posterior distribution obtained in the (n-1)-th auction is used as the prior distribution in the n-th auction. At the very beginning, the prior distribution is assumed to be uniform over the domain, i.e., $\mathbf{f}_{prior}(\theta_j^l) = 1/n_j \;\; \forall\, l = 1 : n_j$.

4.3.2.3 Experimental Study

In this section, we conduct the experiments to validate the performance of the Bayes estimator-based bidding strategy for the target-oriented SLA design in sequential auction . The test environment is setup as follows:

- There are three participating atomic service providers: asp_1, asp_2 and asp_3. In **test case 1**, asp_1 is assumed to employ the Bayes estimator and can learn the type values of asp_2 and asp_3. This learning capability enables asp_1 to propose the bid using the type value-based bidding strategy, i.e., Procedure.4.5. asp_2 and asp_3 do not hold this type of advanced learning capabilities and bid using the self-adaptive

bidding strategy, i.e., Procedure.4.4. In **test case 2**, none of the three atomic service providers holds the capability of Bayes estimator.

- In both test cases, the total number of the sequential auctions is setup as 30. The type value of asp_2 is setup as 1050 and the type value of asp_3 is setup as 1100. With respect to asp_1, we let its type value increase from 1000 to 1200 with interval of 5. This design is to test how the Bayes estimator affects the performance under different parameter settings. For instance, let us consider two parameter configurations of $\{\theta_{asp_1} = 1000, \theta_{asp_2} = 1050, \theta_{asp_3} = 1100\}$ and $\{\theta_{asp_1} = 1200, \theta_{asp_2} = 1050, \theta_{asp_3} = 1100\}$. With the first parameter configuration, asp_1 tends to propose a lower bid due to its smaller type value, thus it is usually chosen as the winner. With the second parameter configuration, asp_1 tends to be always the loser due to its larger type value. This prediction should not be difficult to get when every bidder has the same bidding strategy, i.e., none of them employs the Bayes estimator-based bidding strategy or hold any advantages in the competitive bidding process. This experiment aims to show whether adopting the Bayes estimator-based bidding strategy will help asp_1 even under a hostile environment, like the second parameter configuration.

In Figure.4.10, the horizontal axis is for the type value of asp_1 and the vertical axis is for the corresponding number of times of winning the auction. We can find that the number of times of winning the auction generally follows the descending trend when the type value increases. Even after the type value becomes higher than 1100, it still can win a certain number of auctions. This is due to the reason that, by employing Bayes estimator, asp_i is able to grasp the opportunity when its competitors propose bids being higher than the asp_i's type value. On the other hand, the result of **test case 2** reported in Figure.4.11 shows that asp_1 cannot win any auction once its type value is higher than 1100.

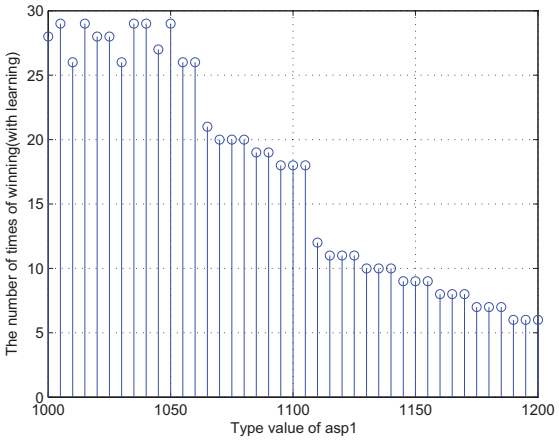

Figure 4.10 Number of times of winning the auction for asp_1(**test case 1**)

The results reported in Figure.4.10 and Figure.4.11 combine to show the effectiveness of employing the Bayes estimator-based bidding strategy for the target-oriented SLA design in sequential auction.

4.3.3 Related Work and Discussion

The research reported in this section are divided into two parts: one-shot auction and sequential auction. Both parts fit to the general scope of the target-oriented SLA design. The one-shot auction can be analyzed using the classical Bayesian game formulation. But the restrictions on the underlying probabilistic distributions limit its applicabilities. We discuss the numerical approaches based on Monte Carlo methods, which is similar to the works reported in [57] and [58]. One of our main contributions lies in the two-dimensional Bidding strategy design using Monte Carlo method and the discussions on the utility maximization strategy based on the competitors' bid pairs.

In the second part of this section, we develop the Bayes estimator-based bidding strategy by exploiting the relationship between the proposed bid rate and type value. This approach depends on some knowledge about the competitors, such as the ratio of

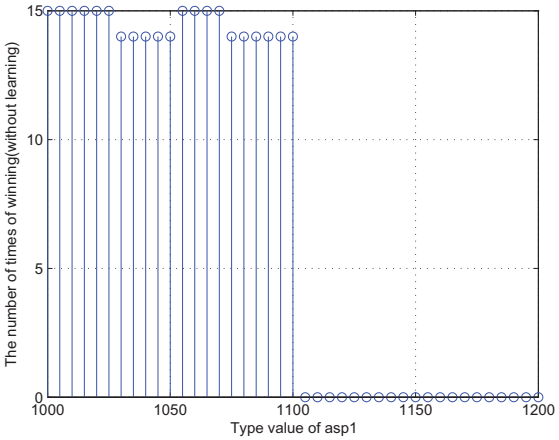

Figure 4.11 Number of times of winning the auction for asp_1(**test case 2**)

the initialization bid to the type value and the descending speed of the bid rate for each auction. Although this information can be captured through statistical estimation methods [59], more detailed models are seeked to be built to capture these unknown factors in an unified framework. We are working on designing strategies to learn the initialization bid, descending speed and type value of an atomic service provider concurrently from its bid rate using a hierarchical model [50].

4.4 Stochastic Modeling of Dynamic Business Relationships

In accordance with the previous discussions, the business relationships between a composite service provider and its controlled atomic service providers can be framed by either bilateral negotiation protocol or auction protocol. The service price specified in the settled SLA varies with the protocol that is used to design the SLA. This service price represents the buying price to the composite service provider and stands for the selling price to the atomic service provider. From the perspective of a composite service provider, the buying price plays an essential role when pricing a composite service which includes that specific atomic service. From the point of view of the atomic service

provider, the selling price represents the revenue that is generated by delivering the related service. Therefore, estimating this service price is important to both sides. In the context of the stochastic performance analysis, the estimation can be represented by the long run probability of framing the business relationship with either one of the SLA design protocols. The modeling approach studied in this section will capture the dynamic interactions of three sides: composite service provider, atomic service provider and service request. This investigation not only helps the composite service provider to estimate the price tag but also provides a controllable approach for the atomic service provider to actively select the SLA design protocol.

4.4.1 Continuous-Time Markov Chains

In Chapter 2, we briefly introduce the discrete-time Markov chains and their applications in the performance modeling of composite services. In this section, we will use continuous-time Markov chain (CTMC), the counterpart of DTMC in the continuous time space, to investigate the performance of the dynamic business relationships framed through either bilateral negotiation protocol or auction protocol.

Comparing to the definition of DTMC, a Markov process is referred to as a CTMC when the parameter space is continuous, which is usually associated with time space in practice. The CTMC dynamics is composed of two types of information: the one-step transition probability from one state to another state, and the average holding time in one state. This dynamics is characterized by the infinitesimal generator matrix $\mathbf{Q}$ [14], and each entry $\mathbf{Q}[i,i]$ is defined in Equation.4.30.

$$\mathbf{Q}[i,j] = \lim_{h \to 0} \frac{\mathbf{Pr}\{X(t+h) = j | X(t) = i\}}{h} \quad \forall\ i \neq j \tag{4.30}$$

The diagonal entry in the $\mathbf{Q}$ is computed through the *in-out balance relationship*, i.e., $\sum_{j=1}^{j=m} \mathbf{Q}[i,j] = 0$. For a homogeneous CTMC, the $\mathbf{Q}[i,j]$ does not vary with parameter t. A CTMC can be transformed into a DTMC through the embedding technique. The

transformed DTMC is called as the embedded DTMC which only captures the probability transition information of the original CTMC [14]. The CTMC analysis can be henceforth conducted on its embedded DTMC. Both a CTMC example and its embedded DTMC are shown in Figure.4.12.

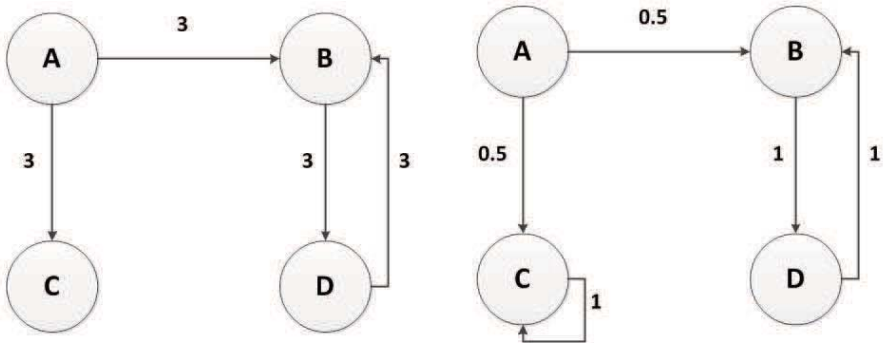

Figure 4.12 A CTMC and its embedded DTMC

4.4.2 The Dynamics of the Business Relationships Framed Through Different SLA Design Protocols

The business relationships connecting a composite service provider and a collection of its controlled atomic service providers can be characterized by a state defined as $\mathbf{S}_i = (a_1, a_2)$, where a_1 represents the number of queued requests for a given type of atomic service and a_2 stands for the number of available atomic service providers. The requests are queued since there exists an upper limit on the number of atomic service providers that a composite service provider can control. Once a service request cannot be immediately served due to the unavailability of atomic service providers, this request will be put into a wait queue owned by the composite service provider. The number of available atomic service providers varies with time. An atomic service provider can identify itself as unavailable when serving a request. After finishing the service provi-

sioning process, this atomic service provider will join the candidate pool. Throughout the life cycle of a service market, the reachable business relationships captures by the states defined this way can be categorized into the following types:

type 1 $\{a_1 > 0$ and $a_2 = 0\}$ corresponds to the scenario where all of the controlled atomic service providers are involved in providing service to the composite service provider through some types of SLA while some requests wait in the queue.

type 2 $\{a_1 = 0$ and $a_2 > 0\}$ corresponds to the scenario where a_2 atomic service providers are available to serve the incoming requests while other atomic service providers are involved in providing service to the composite service provider through some types of SLA.

type 3 $\{a_1 = 0$ and $a_2 = 0\}$ corresponds to the scenario where all of the controlled atomic service providers are involved in providing service to the composite service provider through some types of SLA while the queue is empty.

Figure.4.13 describes the transition dynamics happening among the states of **type 1**. An incoming service request will enable system to transit from state $(i+1, 0)$ to $(i, 0)$. When the system is in state $(i+1, 0)$, a newly available atomic service provider will frame its relationship with the composite service provider through the bilateral negotiation-based SLA design protocol. When we choose the first-come-first-serve scheduling policy [60], the service request lying in the head of the wait queue will be served. State $(1, 0)$ is a boundary state among the states of **type 1**. Its transitions with state $(2, 0)$ are captured in Figure.4.13, while its transitions with another neighboring state $(0, 0)$ are captured in Figure.4.15.

Figure.4.14 describes the transition dynamics happening among the states of **type 2**. A newly available atomic service provider will move the system from state $(0, i)$ to state $(0, i + 1)$. On the other hand, an incoming service request will trigger the system start

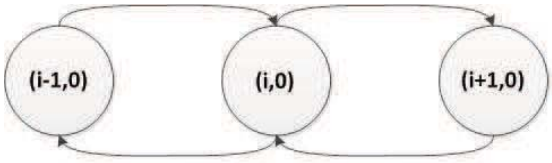

Figure 4.13 The transitions among states of type 1

the auction protocol to determine the atomic service provider which will provide the service to this incoming request. This will cause the system to move from state $(0, i+1)$ to state $(0, i)$. Similar to the case of state $(1, 0)$, state $(0, 1)$ is also a boundary state among the states of **type 2**. Its transitions with state $(0, 2)$ are captured in Figure.4.14, while its transitions with another neighboring state $(0, 0)$ are captured in Figure.4.15.

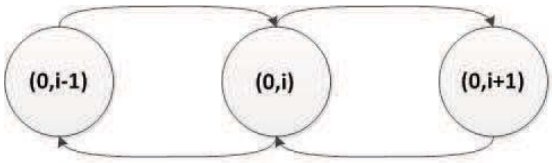

Figure 4.14 The transitions among states of type 2

The state $(0, 0)$ is a boundary state connecting the states of **type 1** and the states of **type 2**. Their relationships are reflected in Figure.4.15. An incoming service request moves the system from state $(0, 0)$ to state $(1, 0)$ of **type 1**. At state $(1, 0)$, a newly available atomic service provider will provide the service to this only existing queued service request after negotiating a SLA with the composite service provider. When system is in state $(0, 1)$, an incoming service request will trigger the negotiation protocol between this available atomic service provider and the composite service provider and move the system to state $(0, 0)$. On the other hand, a newly available atomic service provider will move the system from state $(0, 0)$ to state $(0, 1)$.

In summation, there are five types of transitions connecting these different types of states.

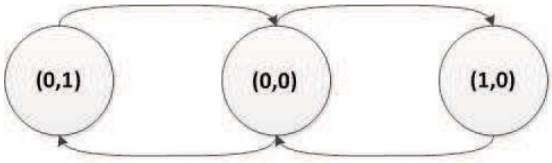

Figure 4.15 The transitions related to state $(0, 0)$

1. The transition from state $(0, i)$ to state $(0, i + 1)$ is triggered by the event that an atomic service provider becomes available.

2. The transition from state $(i + 1, 0)$ to state $(i, 0)$ is triggered by the event that a newly available atomic service provider is assigned to serve the service request lying in the head of the queue through the bilateral negotiation-based SLA design protocol. The auction-based SLA design protocol will not be used here since there only exists a single atomic service provider associated with this type of transition.

3. The transition from state $(i, 0)$ to state $(i + 1, 0)$ is triggered by the event that a service request enters the wait queue.

4. When $i > 1$, the transition from state $(0, i + 1)$ to state $(0, i)$ is triggered by the event that an atomic service provider wins the auction to serve an arriving service request.

5. The transition from state $(0, 1)$ to state $(0, 0)$ is triggered by the event that the only available atomic service provider is assigned to serve the arriving service request through the bilateral negotiation-based SLA design protocol.

The speed of the first type of transition and the speed of the second type of transition are controlled by the service time of the atomic service providers. The speed of the remaining types of transitions is controlled by the interarrival times of the service request. These two facts will be used later to build the CTMC for the dynamic business relationships.

Comparing to the average service time and the interarrival times for the service requests, the computational time required for the negotiation protocol and the auction protocol is considered to be quite small. For instance, at a state of **type 2**, once an atomic service provider becomes available, it will immediately be assigned to serve a queued request after designing the SLA based on bilateral negotiation or auction. Hence, it is impossible for a state belonging to any of the above three types of states to reach a state of the form $(a_1, a_2) : a_1 \geq 1$ and $a_2 \geq 1$. The only possible scenario is that the system starts from the state of $(a_1, a_2) : a_1 \geq 1$ and $a_2 \geq 1$. Once this happens, depending on the relative difference between a_1 and a_2, a serious of instantaneous transitions will be invoked and a set of associated instantaneous states will be constructed. Therefore, the system can be thought of starting from $(0, a_2 - a_1), (0, 0)$, and $(a_1 - a_2, 0)$ respectively. The analysis is illustrated in Figure.4.16.

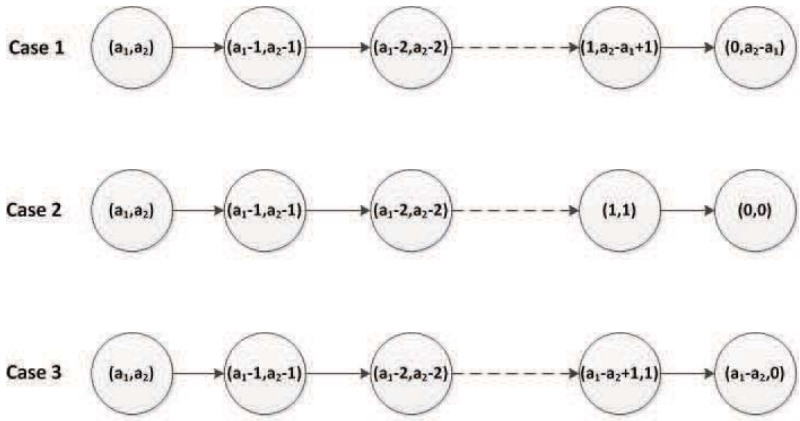

Figure 4.16 Three types of evolving processes consisting of instantaneous states

4.4.3 CTMC-Based Performance Analysis

In accordance with the state and transition definitions, Figure.4.17 shows the Markov chain model built for capturing the dynamic business relationships connecting a composite service provider and a set of its controlled atomic service providers, which are

further affected by incoming service requests. In this example, the size of the service wait queue and the number of the controlled atomic service providers are assumed to be five, which determines the size of the constructed Markov chain. The arrival process of the incoming service request is assumed to follow the Poisson process with the arrival rate of λ; and the service time for each atomic service provider is assumed to be an exponential random variable with rate μ. Based on both λ and μ, the rate associated with each transition is calculated and marked in Figure.4.17. The concrete computation process are given as follows:

- The transitions from state $(0, i)$ to state $(0, i-1)$, from state $(0, 0)$ to state $(1, 0)$, and from state $(i, 0)$ to state $(i+1, 0)$ are triggered by an incoming service request which has a rate of λ.

- The fact that the system is in state $(i, 0)$ indicates that all of the five atomic service providers are involved in the service provisioning process. Based on the property of the superposition of Poisson processes [15], the associated rate for making transition from $(i, 0)$ to $(i-1, 0)$ and from $(0, 0)$ to $(0, 1)$ is 5μ.

- The fact that the system is in state $(0, i)$ indicates that $5-i$ atomic service providers are involved in the service provisioning process. Therefore, the associated rate for making transition from $(0, i)$ to $(0, i+1)$ is $(5-i)\mu$.

Once a Markov chain model is constructed, the numerical approach or the analytical approach to get the stationary probability distribution is a standard procedure. Please refer to [14] and [15] for more details. Our research has two focuses. The first one is about how to build the Markov chain model based on the analysis of the system dynamics, and has been studied in the above. The second one aims to investigate which kind of performance metrics can be derived from the constructed Markov model to help both the composite service provider and the atomic service provider in their respective managerial decision making process. This focus will be discussed in the following part.

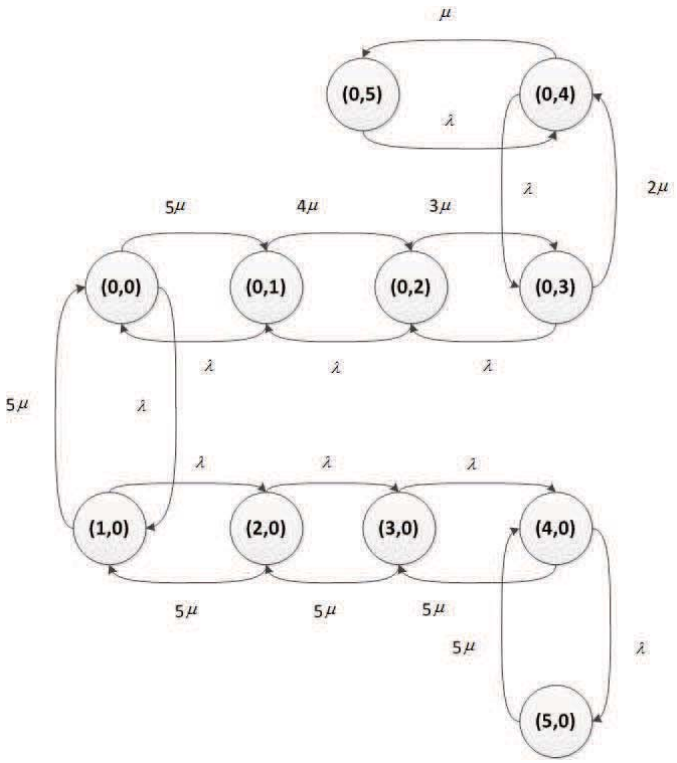

Figure 4.17 Markov chain model for the dynamic business relationships framed through different SLA design protocols

From the viewpoint of a composite service provider, a related performance metric is:

- What is the expected price tag for this type of atomic service which is bought from its controlled atomic service providers?

The composite service provider needs this information to compute the price charged to the consumer over a composite service request which involves this specific atomic service. The estimation of the price can be computed as in Equation.4.31.

$$\mathbf{E}[P] = \sum_{i=1}^{i=5} \mathbf{EP}_n\{(i,0)\} \times \mathbf{Pr}(i,0) + \sum_{i=2}^{i=5} \mathbf{EP}_a\{(0,i)\} \times \mathbf{Pr}(0,i) + \mathbf{EP}_n\{(0,1)\} \times \mathbf{Pr}(0,1) \tag{4.31}$$

In Equation.4.31, $\mathbf{EP}_n\{(i,0)\}$ represents the expected price tag specified by the bilateral negotiation-based SLA design initiated at state $(i,0)$; $\mathbf{EP}_a\{(0,i)\}$ represents the

expected price tag specified by the auction-based SLA design initiated at state $(0, i)$. Note that the state $(0, 1)$ corresponds to the negotiation protocol rather than the auction protocol due to the existence of only one atomic service provider. Another issue in using Equation.4.31 is the computation of $\mathbf{Pr}(i, 0)$ and $\mathbf{Pr}(0, i)$. The Markov chain analysis outputs the stationary probability which represents the long run proportion of staying in a given state. Let $\pi\{(i, j)\}$ denote the stationary probability of state (i, j). The stationary probability of $\pi\{(i, j)\}$ is obtained by solving the system of global balance equations [15], which are just standard procedures once the Markov model has been built. However, we can not simply use $\pi\{(i, j)\}$ to replace $\mathbf{Pr}\{(i, j)\}$. For instance, at state $(0, i)$, the system can move to either state $(0, i + 1)$ or state $(0, i - 1)$. Only the transition leading to state $(0, i - 1)$ corresponds to the event of invoking the auction protocol. This portion of $\pi\{(0, i)\}$ to be allocated to $\mathbf{Pr}(0, i)$ is controlled by the probability of having a service request arriving earlier than a newly available atomic service provider. Similarly, for a state of $(i, 0)$, the proportion of invoking a bilateral negotiation protocol is controlled by the relative speed between the arrival of the atomic service provider and the arrival of another service request. If an atomic service provider arrives earlier, a negotiation protocol will be invoked and leave the state of $(i, 0)$. In summation, we have

$$\begin{aligned} \mathbf{Pr}\{(0, i)\} &= \frac{\lambda}{\lambda + (5 - i)\mu}\pi\{(0, i)\} \quad i = 2, 3, 4 \\ \mathbf{Pr}\{(i, 0)\} &= \frac{5\mu}{\lambda + 5\mu}\pi\{(i, 0)\} \quad i = 1, 2, 3, 4 \end{aligned} \tag{4.32}$$

There are three exceptional states: $(0, 0)$, $(5, 0)$ and $(0, 5)$. State $(0, 0)$ does not initiate any SLA design protocols and is not included in Equation.4.31. There exists only one output transition for both $(5, 0)$ and $(0, 5)$. Hence, $\mathbf{Pr}(5, 0) = \pi\{(5, 0)\}$ and $\mathbf{Pr}(0, 5) = \pi\{(0, 5)\}$. Besides the probabilities of $\mathbf{Pr}\{(0, i)\}$ and $\mathbf{Pr}\{(i, 0)\}$ in Equation.4.31, the computation of the expected price $\mathbf{EP}_n\{(i, 0)\}$ and $\mathbf{EP}_a\{(0, i)\}$ is another issue to be addressed. In accordance with the previous discussions on the ne-

gotiation protocol design and auction protocol design, the specified price tags heavily rely on the concrete implementations. In practice, the general approach of getting this kind of estimation is to do the density estimation based on the accumulated transaction history records using either nonparametric or parametric method [61][62]. The estimated density function enables us to compute the expected price tags and other necessary distribution descriptors, like variance. This type of approach is operational and performance tunable based on the quality of the accumulated data and the applied algorithms.

An atomic service provider can have three different types of statuses: in the service provisioning process, in the candidate pool and off the market. The status of "off the market" can be due to a variety of reasons, e.g., maintenance, or terminate the business relationship with the composite service provider,etc. In the current research setting, we do not consider the "off the market" status and assume that the atomic service provider is either in the service provisioning process or in the candidate pool waiting to be invoked by the composite service provider. From the perspective of an atomic service provider, two related performance metrics are listed as follows:

1. What is the long run probability of landing in a scenario of framing the SLA configuration with the composite service provider through the negotiation protocol?

2. What is the long run probability of landing in a scenario of framing the SLA configuration with the composite service provider through the auction protocol?

These two performance metrics have managerial meanings to the atomic service provider. If it is very likely for an atomic service provider to get involved into a bilateral negotiation-based SLA design protocol, then this atomic service provider do not need to worry too much about the competition. This can bring more bargaining power into its relationship with the composite service provider. Moreover, some strategic decisions can be made based on these two performance metrics. In practice, developing successful strategy for

either the auction process or the bilateral negotiation process can cost a lot of resources, both computational ones and social ones. Based on these two performance metrics, an atomic service provider can decide which strategy should be invested in.

Let $\mathcal{Q}_n$ denote the probability of landing in a negotiation protocol and $\mathcal{Q}_a$ denote the probability of landing in an auction protocol. We have:

$$\mathcal{Q}_n = \frac{\sum_{i=1}^{i=5} \mathbf{Pr}\{(i,0)\} + \mathbf{Pr}\{(0,1)\}}{\sum_{i=1}^{i=5} \pi\{(i,0)\} + \pi\{(0,0)\} + \sum_{i=1}^{i=5} \pi\{(i,0)\}} \tag{4.33}$$

In Equation.4.33, the physical meaning of $\mathbf{Pr}\{(i,0)\}$ and its computation has been discussed in the above. The denominator for the right side of the equation is the sum of the stationary probability vector and equals to 1. Therefore,

$$\mathcal{Q}_n = \sum_{i=1}^{i=5} \mathbf{Pr}\{(i,0)\} + \mathbf{Pr}\{(0,1)\}. \tag{4.34}$$

Similarly, we have

$$\mathcal{Q}_a = \sum_{i=2}^{i=5} \mathbf{Pr}\{(0,i)\}. \tag{4.35}$$

4.4.4 Related Work and Discussion

A Markov chain model is constructed to model an auction-based logistics market [63]. Instead of focusing on a single type of business relationship framing protocol, our research focuses on multiple business relationship framing protocols, e.g., bilateral negotiation-based SLA design and the auction-based SLA design. From the performance analysis perspective, the research presented in [63] aims to analyze the effects of the system parameters on the profit of different carriers managed by a single type of business relationship. On the other hand, we are interested in developing analytical approaches to help the decision making processes of both composite service provider and atomic service providers which are involved in a set of dynamic business relationships.

In the example shown in Figure.4.17, the size of the wait queue and the number of the maximal available atomic service providers are assumed to be five. Once this

number is increased for complex systems, the constructed Markov chain will become very complicated and even cannot be drawn explicitly, which is related to the issue of state explosion [64]. To meet this challenge, we need research the application of high level formalism, such as Petri nets [65], to model the dynamic business relationships in a more abstract way. The current model assumes that every atomic service service provider shares the same service rate. Taking into account the heterogeneous service capability of different atomic service providers is another extension that should be made for the current model construction process.

CHAPTER 5. Conclusions, Discussion and Future Work

Our research focuses on three facets of competitive service market: modeling, storage design and SLA-based management. In this chapter, we conclude this dissertation with a summary of the achieved results and the contributions to establishing a manageable and economically feasible service market. We also briefly summarize directions of future work at the end of this chapter.

5.1 Conclusions

We have proposed and formalized the concept of a competitive service market, which is created to capture the business dynamics generated by various types of service agents interconnected under the framework of services computing. The structural hierarchy consisting of atomic service provider and composite service provider clarifies their respective positions held in the market. The supply and demand relationships among service consumers, composite service providers and atomic service providers frame the interconnections of the service chain in the service market. These complexities are instantiated through a variety of constraints: functionality constraint, QoS constraint and profit constraint.

In order to provide quantitative support for the decision making processes of different market players, we have constructed stochastic models to conduct performance analysis at different levels spanning vertically on the structural hierarchy of the service market. The queuing model is used as a unified approach to model the dynamic behaviors at the

atomic service level. We have demonstrated how the fundamental performance metrics are used to derive the compound metrics. The end-to-end response time for composite service is one of the most widely studied performance metrics in services computing. We have introduced the time-dependent system dynamics and discuss its effect on the response time computation. We have created a recursive procedure that keeps track of the time difference between the starting time of the composite service and the starting time of a participating atomic service. Using market-based availability as an example, we have investigated the stochastic modeling approach at the market level. This line of research not only creates a systematic analysis methodology which is multi-level and metrics-dependent, but also delivers a set of operational solutions to assist various types of decision makers in the market.

To gain a competitive edge in the service market, a service provider should be able to deliver its service with diverse quality levels based on the requirements of the service purchaser. In the context of service requirements analysis, we capture this fact and classify the concept of service and the concept of service parameter configuration in terms of functional and non-functional features. The concept of service focuses on the functionalities and the concept of service parameter configuration centers around the non-functional features. Moreover, we have structured the service discovery process into a two-step procedure: first, to find the candidate that meets the functional requirements of the customer; secondly, to determine the appropriate service parameter configuration that fulfills the non-functional requirements. It is required to save both candidate services and candidate service parameter configurations in the service registry and keep track of their chaining relationships.

The combinatorial relationship between a single service and its associated service parameter configurations leads us to propose a scalable storage architecture that can handle the large amount of service parameter configurations while supporting the two-stage service discovery process. The designed architecture is composed of a set of decentralized

storage units indexed by a centralized service array. We apply the counting Bloom filter to design the storage unit and it has two features:

- The counting Bloom filter-based set membership evaluation criterion enables us to block the service request which cannot find the candidate service parameter configuration in a very efficient way.
- The collection of candidate service parameter configurations is decomposed into a set of linked lists, and the length of each linked list is monitored by the counter value of the counting Bloom filter.

These two features are combined to minimize the number of comparison operations, which determines the computational cost of this two-stage service discovery process. We have developed an analytically solvable metrics, expected counter value in the counting Bloom filter, to predict the performance of this system. Since this metric is a function of the system parameter settings, the system performance is thus tunable by changing system parameters. This feature is helpful to build a controllable system that is essential for the system manageability.

The business relationships of different market players are typically governed by the service level agreements (SLAs). We systematically investigate the SLA design methodologies for every identified SLA design pattern. In order to capture the behavior of either the negotiation counterpart or the market competitors, we develop Bayes estimator-based scheme for the SLA design protocols in both the bilateral negotiation pattern and sequential auction pattern. With the help of the learning capability provided by the Bayes estimator, we are able to develop flexible offer proposition and bidding strategies in accordance with different SLA settlement criteria. The experimental results demonstrate its effectiveness. Instead of considering each SLA design instance as an independent event, we cast the dynamic business relationships framed through coexisting SLA design protocols into a unified stochastic model based on Markov chain. We study

a variety of performance metrics that can be analyzed using the constructed Markov chain model, and how these performance metrics can help different market players to make both strategic and operational decisions.

In summary, the proposition of the concept of a competitive service market and the involved interrelations enhance the business dimension of services computing, and provide a framework to study the business behaviors of various types of service agents. The modeling approaches not only deliver operational solutions to some important performance metrics but also help to build a multi-level modeling framework. Guided by the service requirements analysis, we design a scalable service storage architecture to support the fast service discovery process while being able to keep track of large number of service parameter configurations. We have systematically investigated the SLA design methodologies based on diverse business scenarios in the life cycle of a competitive service market. The CTMC-based modeling approach captures the dynamic business relationships in a unified system that can be quantitatively analyzed. These three facets of our research are combined to form a foundation towards building a service management system of the competitive service market.

5.2 Discussion: Service Demand Model in the Competitive Service Market

In the current research setting, we assume that the payment scheme is "pay-per-service instance". Each invocation of an atomic service costs the composite service provider a fixed payment which is specified in the signed SLA. Once the pair of unit payment and quality level is fixed, the composite service provider will route the incoming atomic service request to this atomic service provider in accordance with a pre-determined service selection protocol. The estimated number of invocation on an atomic service provider is modeled as a demand function which takes the retail price

and quality levels as the input parameters [66]. Here, the retail price is the one that the composite service provider charges the consumer over this specific atomic service. Equation.5.1 gives a generic form of demand function.

$$M_i = \mathcal{D}_i(P_0, Q_i) \tag{5.1}$$

In Equation.5.1, the demand on the service provisioned by asp_i is represented as a function of P_0 and Q_i. Q_i denotes the quality level specified in the SLA signed between asp_i and csp_o. P_0 stands for the retail price that csp_o will ask. Hence, $P_0 - P_i$ represents the net profit for csp_o by purchasing this specific atomic service for the scheduled composite service. Here, P_i denotes the selling price specified in the SLA. In practice, empirical study and statistical approaches are applied to construct the concrete demand functions [67]. The demand function can be directly used to compute the utility function in a profit-driven business environment. The utility of an atomic service provider, asp_i, is computed as

$$\mathcal{U}_{asp_i} = \mathcal{D}_i(P_0, Q_i) \times P_i - \mathcal{C}_i(Q_i) \tag{5.2}$$

On the other hand, the utility function of the composite service provider by purchasing this atomic service is given as follows.

$$\mathcal{U}_{csp_o} = P_0 \times \mathcal{D}_i(P_0, Q_i) - P_i \times \mathcal{D}_i(P_0, Q_i) \tag{5.3}$$

In Equation.5.2, $\mathcal{C}(Q_i)$ represents the service delivery cost required to maintain the pre-specified quality level of Q_i, where the exact formula of $\mathcal{C}_i(\cdot)$ depends on a variety of factors, such as business capability of atomic service provider and the underlying service infrastructure. For instance, the business operation cost is claimed to be able to be modeled by the quadratic form of the quality level [68], e.g., $\frac{1}{2}\varsigma_i Q_i^2$. Here, ς_i is a constant and referred to as the *quality level coefficient*.

Explicitly defining the demand function enables us to formulate the utility in a well structured function like Equation.5.2. However, we need to stress the fact that fulfill-

ing this task heavily depends on the demand function which needs advanced modeling approach and accumulated transaction data to get its accurate formula [69].

5.3 Future Work: Sequential SLA Design Using Stackelberg Game

The SLA design scenarios investigated so far do not explicitly take other existing SLAs into account. This simplification does not fully expose the complicated business scenarios that a SLA design process can get involved in. A typical scenario is that an atomic service provider designs SLA together with the composite service provider. After observing this signed SLA, another atomic service provider decides how to design its SLA configuration based on the observed information. These two sequential SLA designs can be modeled as a Stackelberg game [54]. To differentiate with the SLA design patterns discussed previously, we refer to this newly proposed pattern as the *sequential SLA design*. In the Stackelberg game, the agents are classified as a leader and a follower. The leader moves first and makes a set of decisions which are observable to the follower which will make its decision based on this observation. With respect to the sequential SLA design, the firstly signed SLA functions as the leader while the secondly signed SLA functions as the follower [54].

As a preliminary modeling effort, we limit the number of involved atomic service providers as two. The SLA firstly signed between asp_1 and csp_o is denoted as SLA_1. The other SLA secondly signed between asp_2 and csp_o is denoted as SLA_2. A signed SLA binds two agents. We can use a weighted sum of participating agents' utilities as the utility for a SLA.

$$\begin{aligned}\mathcal{U}_1(P_{0,1}, P_1, P_2, Q_1, Q_2) &= w_1(P_{0,1} - P_1)\mathcal{D}_1^c(P_{0,1}, P_{0,2}, Q_1, Q_2) \\ &\quad + w_2(P_1\mathcal{D}_1^c(P_{0,1}, P_{0,2}, Q_1, Q_2) - \mathcal{C}_1(Q_1)) \qquad (5.4)\end{aligned}$$

Equation.5.4 computes the utility for the firstly designed SLA, SLA_1. It takes the payoffs of both sides into account. We use w_1 and w_2 to represent the different weights held by two counterparts. In Equation.5.4, $(P_{0,1} - P_1)\mathcal{D}_1^c(P_{0,1}, P_{0,2}, Q_1, Q_2)$ represents the payoff of csp_o while $(P_1\mathcal{D}_1^c(P_{0,1}, P_{0,2}, Q_1, Q_2) - \mathcal{C}_1(Q_1))$ represents the payoff of asp_1. Here, $P_{0,1}$ represents the csp_o's retail price of the service that is purchased from asp_1 through SLA_1. The specification of SLA_1 determines the quality level of the service and incurs a demand on this particular service parameter configuration. The demand on the service signed through SLA_1 not only depends on its retail price and the specified quality level, i.e., $\{P_{0,1}, Q_1\}$, but also depends on the competitive business relationship framed by SLA_2. For instance, a liner demand function to capture the competitor's effect is given in Equation.5.5 [70].

$$\mathcal{D}_1^c(P_{0,1}, P_{0,2}, Q_1, Q_2) = \alpha - \theta_1 \times (P_{0,1} - P_{0,2}) + \theta_2 \times (Q_1 - Q_2) \tag{5.5}$$

In Equation.5.5, a linear factor of θ_1 measures the effect of price difference and a linear factor of θ_2 measures the effect of quality level difference, α is a constant. Similar to the SLA_1, the utility of SLA_2 is given in Equation.5.6.

$$\begin{aligned} U_2(P_{0,2}, P_1, P_2, Q_1, Q_2) &= w_1(P_{0,2} - P_2)\mathcal{D}_2^c(P_{0,2}, P_{0,1}, Q_1, Q_2) \\ &+ w_2(P_2\mathcal{D}_2^c(P_{0,2}, P_{0,1}, Q_1, Q_2) - \mathcal{C}(Q_2)) \end{aligned} \tag{5.6}$$

Following the backward solution approach for solving Stackelberg game [54], the specification of SLA_2 is solved through

$$\begin{aligned} P_{0,2}^* &= \text{argmax}\{\mathcal{U}_2(P_{0,2}, P_2^*, Q_2^*, |P_{0,1}, P_1, Q_1\} \quad \forall P_{0,2} \in \mathcal{P}_{0,2} \\ P_2^* &= \text{argmax}\{\mathcal{U}_2(P_{0,2}^*, P_2, Q_2^*, |P_{0,1}, P_1, Q_1\} \quad \forall P_2 \in \mathcal{P}_2 \\ Q_2^* &= \text{argmax}\{\mathcal{U}_2(P_{0,2}^*, P_2^*, Q_2, |P_{0,1}, P_1, Q_1\} \quad \forall Q_2 \in \mathcal{Q}_2 \end{aligned} \tag{5.7}$$

The equilibrium solution for the specification of SLA_2 is obtained by assuming the specification of SLA_1, and is solved in Equation.5.7. The results of Equation.5.7 imply

that $P^*_{0,2}$,P^*_2 and Q^*_2 are the functions of $P^*_{0,1}$,P^*_1 and Q^*_1. Hence, we have

$$\begin{aligned} P^*_{0,1} &= \text{argmax}\{\mathcal{U}_1(P_{0,1}, P^*_1, Q^*_1\} \quad \forall P_{0,1} \in \mathcal{P}_{0,1} \\ P^*_1 &= \text{argmax}\{\mathcal{U}_1(P^*_{0,1}, P_1, Q^*_1\} \quad \forall P_1 \in \mathcal{P}_1 \\ Q^*_1 &= \text{argmax}\{\mathcal{U}_1(P^*_{0,1}, P^*_1, Q_1\} \quad \forall Q_1 \in \mathcal{Q}_1 \end{aligned} \tag{5.8}$$

In the above equations, $\mathcal{P}_{0,2}, \mathcal{P}_2, \mathcal{Q}_2, \mathcal{P}_{0,1}, \mathcal{P}_1$ and $\mathcal{Q}_1$ are the domains of the corresponding variables. The above discussions just provide preliminary analysis of the sequential SLA design using Stackelberg game formulation. In practice, solving Eq.5.7 and Eq.5.8 presents real challenges since the demand functions, $\mathcal{D}^c_1$ and $\mathcal{D}^c_2$ are not differentiable. Some researches have been reported to numerically solve the real-world Stackelberg games and can be applied to solve our problem [71][72].

We have discussed several SLA design patterns in both Chapter 4 and this section. Future research would also include developing a generic service management console that is able to instantiate a specific SLA design protocol in accordance with a given business scenario. This requires a systematic decision making process to help selecting which SLA design protocol should be used. The SLA design methodologies developed in the above will constitute the candidate pool for this decision making process.

Bibliography

[1] Y.Huang and J.-Y. Chung, "A Web Services-based Framework for Business Integration Solutions," *Electronic Commerce Research Applications*, vol. 2, pp. 15–26, 2003.

[2] Q. C. T. Hsing Kenneth Cheng and J. Zhao, "Web services and service-oriented application provisioning: An analytical study of application servcie strategies," *IEEE Transactions on Engineering Management*, vol. 53, pp. 520–533, 2006.

[3] S. Cheng, C. K.Chang, L. J. Zhang, and T. H.Kim, "Towards Competitive Web Service Market." in *Proceedings of the IEEE International Workshop on Future Trends of Distributed Computing Systems(FTDCS'07)*, 2007, pp. 213–219.

[4] L. J. Zhang and B. Li, "Requirements Driven Dynamic Services Composition for Web Services and Grid Solutions," *Journal of Grid Computing*, vol. 2, no. 2, pp. 121–140, 2004.

[5] D. Ardagna and B. Pernici, "Adaptive service composition in flexible processes," *IEEE Transactions on Software Engineering*, vol. 33, no. 6, pp. 369–384, 2007.

[6] R. Berbner, M. Spahn, N. Repp, O. Heckmann, and R. Steinmetz, "Heuristics for QoS-aware Web Service Composition." in *Proceedings of 2006 IEEE International Conference on Web Services (ICWS 2006)*, September 2006, pp. 72–82.

[7] L. J. Zhang, S. Cheng, Y.-M. Chee, A. Allam, and Q. Zhou, "Pattern Recognition Based Adaptive Categorization Technique and Solution for Services Selection." in

Proceedings of the 2nd IEEE Asia-Pacific Service Computing Conference, 2007, pp. 535–543.

[8] S. Cheng, C. K. Chang, and L.-J. Zhang, "An Efficient Service Discovery Algorithm for Counting Bloom Filter-Based Service Registry," in *Proceedings of the IEEE International Conference on Web Services (ICWS'09)*, 2009, pp. 157–164.

[9] S. Cheng, C. K.Chang, and L. J. Zhang, "Stochastic Modeling Study for Competitive Web Services Market." in *Proceedings of the IEEE International Conference on Web Services(ICWS'07)*, 2007, pp. 960–967.

[10] P. O'Connor, *Practical Reliability Engineering.* Wiley, 2002.

[11] J. A. Buzacott and J. G. Shanthikumar, *Stochastic Models of Manufacturing Systems.* Prentice Hall, 1993.

[12] L. Kleinrock, *Queueing Systems. Volume 1: Theory.* Wiley-Interscience, 1975.

[13] D. W. Stroock, *An Introduction to Markov Processes.* Springer, 2005.

[14] W. J. Stewart, *Introduction to the Numerical Solution of Markov Chains.* Princeton University Press, 1994.

[15] W. J.Stewart, *Probability, Markov Chains, Queues, and Simulation.* Princeton University Press, 2009.

[16] C. Ghezzi, D. Mandrioli, S. Morasca, and M. Pezzè, "A Unified High-Level Petri Net Formalism for Time-Critical Systems," *IEEE Transactions on Software Engineering*, vol. 17, pp. 160–172, 1991.

[17] B. Berthomieu and M. Diaz, "Modeling and Verification of Time Dependent Systems Using Time Petri Nets," *IEEE Transactions on Software Engineering*, vol. 17, pp. 259–273, 1991.

[18] T. Erl, *SOA Principles of Service Design*. Prentice Hall, 2008.

[19] M. P. Papazoglou and W.-J. Heuvel, "Service oriented architectures: Approaches, technologies and research issues," *The VLDB Journal*, vol. 16, pp. 389–415, 2007.

[20]

[21] G. Myers, "A Fast Bit-vector Algorithm for Approximate String Matching Based on Dynamic Programming," *Journal of the ACM*, vol. 46, pp. 395–415, 1999.

[22] D. Gusfield, *Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology*. Cambridge University Press, 1997.

[23] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, and S. Weerawarana, "Unraveling the Web Services Web: An Introduction to SOAP, WSDL, and UDDI," *IEEE Internet Computing*, vol. 6, no. 2, pp. 86–93, Mar/Apr 2002.

[24] B.Bloom, "Space/time tradeoffs in hash coding with allowable errrors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[25] A. Broder and M. Mitzenmacher, "Network Applications of Bloom Filters: A Survey," *Internet Mathematics*, vol. 1, no. 4, pp. 485–509, 2002.

[26] S. S. Skiena, *The Algorithm Design Manual*. Springer, 2010.

[27] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

[28] L. Fan, P. Cao, J. Almeida, and A. Z. Broder, "Summary cache: a scalable wide-area web cache sharing protocol," *IEEE/ACM Trans. Netw.*, vol. 8, no. 3, pp. 281–293, 2000.

[29] H. Song, S. Dharmapurikar, J. Turner, and J. Lockwood, "Fast Hash Table Lookup Using Extended Bloom Filter: An Aid to Network Processing," in *SIGCOMM*, 2005, pp. 181–192.

[30] P. A. MacMahon, *Combinatory analysis*. Dover Publications, 2004.

[31] H.Song, S. Dharmapurikar, J. Turner, and J.Lockwood, "Fast Hash Table Lookup Using Extended Bloom Filter: An Aid to Network Processing." in *SIGCOMM*, 2005, pp. 181–192.

[32] D. Guo, J. Wu, H. Chen, and X. Luo, "Theory and Network Applications of Dynamic Bloom Filter," in *INFOCOM*, 2006, pp. 1–12.

[33] R. Chakravorty, I. Pratt, and J. Crowcroft, "A Framework for Dynamic SLA-based QoS Control for UMTS," *IEEE Wireless Communications*, vol. 10, no. 5, pp. 30–37, 2003.

[34] H. Song, A. Banerjee, B. Mukherjee, B.-W. Kim, S. Yang, and Y. Park, "SLA-Aware Protocol for Efficient Tunable Laser Utilization to Support Incremental Upgrade in Long-Reach Passive Optical Networks," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 1, no. 5, pp. 512–520, 2009.

[35] T. Ahmed, A. H. Asgari, A. Mehaoua, E. Borcoci, L. Berti-íquille, and K. Georgios, "End-to-End Quality of Service Provisioning Through an Integrated Management System for Multimedia Content Delivery," *Computer Communications*, vol. 30, no. 3, pp. 638–651, 2007.

[36] K. Xiong and H. Perros, "SLA-based Service Composition in Enterprise Computing." in *Proceedings of the 17th IEEE International Workshop on Quality of Service*, 2008, pp. 30–39.

[37] J. Yan, R. Kowalczyk, J. Lin, M. B. Chhetri, S. K. Goh, and J. Zhang, "Autonomous Service Level Agreement Negotiation for Service Composition Provision," *Future Generation Computer Systems*, vol. 23, no. 6, pp. 748–759, 2007.

[38] T. Taleb., K. Hashimoto, N. Kato, and Y. Nemoto, "A Dynamic Service Level Negotiation Mechanism for QoS Provisioning in NGEO Satellite Networks," in *Proceedings of the IEEE International Conference on Communications (ICC'07)*, 2007, pp. 1–6.

[39] M. Perry and H. Kaminski, "SLA Negotiation System Design Based on Business Rules," in *Proceedings of the IEEE International Conference on Services Computing (SCC'08)*, 2008, pp. 609–612.

[40] A. Dan, D. Davis, R. Kearney, A. Keller, R. King, D. Kuebler, H. Ludwig, M. Polan, M. Spreitzer, and A. Youssef, "Web Services on Demand: WSLA-driven Automated Management," *IBM Systems Journal*, vol. 43, no. 1, pp. 136–158, 2004.

[41] M. Comuzzi and B. Pernici, "A Framework for QoS-based Web Service Contracting," *ACM Transactions on the Web*, vol. 3, no. 3, pp. 1–52, 2009.

[42] H. Chuan, G. Lei, B. Du, Z. Huang, and S. Li, "A WSLA-Based Monitoring System for Grid Service-GSMon," in *Proceedings of the IEEE International Conference on Services Computing(SCC'04)*, 2004, pp. 596–599.

[43] R. B. Myerson and M. A. Satterthwaite, "Efficient Mechanisms for Bilateral Trading," *Journal of Economic Theory*, vol. 29, pp. 265–281, 1983.

[44] C. Fox, *Introduction to Software Engineering Design: Processes, Principles and Patterns with UML2*. Addison Wesley, 2006.

[45] V. Krishna, *Auction Theory*. Academic Press, 2009.

[46] J. Bulow and P. Klemperer, "Auctions versus Negotiations," *American Economic Review*, vol. 86, pp. 180–194, 1993.

[47] G. Casella and R. L. Berge, *Statistical Inference (2nd Edition)*. Duxbury Press, 2001.

[48] S. M. Lynch, *Introduction to Applied Bayesian Statistics and Estimation for Social Scientists*. Springer, 2007.

[49] K. B. Athreya and S. N. Lahiri, *Measure Theory and Probability Theory*. Springer, 2006.

[50] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian Data Analysis*. Chapman and Hall/CRC, 2003.

[51] C. Li, J. Giampapa, and K. Sycara, "Bilateral Negotiation DecisionsWith Uncertain Dynamic Outside Options," *IEEE Transactions on Systems, Many, and Cybernetics, Part C: Applications and Reviews*, vol. 36, no. 1, pp. 31–44, 2006.

[52] D. Zeng and K. Sycara, "Bayesian learning in negotiation," *International Journal of Human-Computer Studies*, vol. 48, no. 1, pp. 125–141, 1998.

[53] K. M. Sim, Y. Guo, and B. Shi, "BLGAN: Bayesian Learning and Genetic Algorithm for Supporting Negotiation With Incomplete Information," *IEEE Transactions on Systems, Many, and Cybernetics, Part B: Cybernetices*, vol. 39, no. 1, pp. 198–211, 2009.

[54] R. B. Myerson, *Game Theory: Analysis of Conflict*. Harvard University Press, 1997.

[55] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*. Springer, 2010.

[56] S. S.Oren and M. H.Rothkopf, "Optimal Bidding in Sequential Auctions," *Operations Research*, vol. 23, pp. 1080–1090, 1975.

[57] V. Gountis and A. Bakirtzis, "Bidding Strategies for Electricity Producers in a Competitive Electricity Marketplace," *IEEE Transactions on Power Systems*, vol. 19, pp. 356–365, 2004.

[58] F. Wen and A. David, "Optimal Bidding Strategies and Modeling of Imperfect Information Among Competitive Generators," *IEEE Transactions on Power Systems*, vol. 16, pp. 15–21, 2001.

[59] T. Li, I. Peffigne, and Q. Vuong, "Semiparametric Estimation of the Optimal Reserve Price in First-Price Auctions," *Journal of Business and Economic Statistics*, vol. 21, pp. 53–64, 2003.

[60] M. L. Pinedo, *Scheduling: Theory, Algorithms, and Systems*. Springer, 2008.

[61]

[62] J. D. Gibbons, *Nonparametric Statistics: An Introduction*. Sage Publications, Inc, 1992.

[63] S. AgralI, B. Tan, and F. Karaesmen, "Modeling and Analysis of an Auction-based Logistics Market," *European Journal of Operational Research*, vol. 191, pp. 272–294, 2008.

[64]

[65] T. Murata, "Petri nets: Properties, Analysis and Applications," *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541 –580, 1989.

[66] A. O'Sullivan and S. M. Sheffrin, *Economics: Principles in Action*. Pearson Prentice Hall, 2006.

[67] S. Shimizua, R. Rangaswamib, H. A. Duran-Limonc, and M. Corona-Perezc, "Platform-independent modeling and prediction of application resource usage characteristics," *Journal of Systems and Software*, vol. 82, no. 12, pp. 2117–2127, 2009.

[68] D. R and S.Moorthy, "Managing a Distribution Channel Under Asymmetric Information With Performance Requirements," *Management Science*, vol. 43, pp. 1628–1644, 1997.

[69] T. Raykov and G. A. Marcoulides, *A First Course in Structural Equation Modeling*. Lawrence Erlbaum Associates, 2006.

[70] A. A. Tsay and N. Agrawal, "Channel dynamics under price and service competition," *Manufacturing and Service Operations Management*, vol. 2, no. 4, pp. 372–391, 2000.

[71] P. Paruchuri, J. P. Pearce, J. Marecki, M. Tambe, F. Ordonez, and S. Kraus, "Efficient Algorithms to Solve Bayesian Stackelberg Games for Security Applications," in *Proceedings of the 23rd national conference on Artificial intelligence*, 2008, pp. 1559–1562.

[72] T. Sandholm, A. Gilpin, and V. Conitzer, "Mixed-integer Programming Methods for Finding Nash Equilibria," in *Proceedings of the 20th national conference on Artificial intelligence*, 2005, pp. 495–501.