3-12-2020

An Approach of QoS Evaluation for Web Services Design With Optimized Avoidance of SLA Violations

Peng Cheng
University of Windsor

Follow this and additional works at: https://scholar.uwindsor.ca/etd

Recommended Citation

This online database contains the full-text of PhD dissertations and Masters' theses of University of Windsor students from 1954 forward. These documents are made available for personal study and research purposes only, in accordance with the Canadian Copyright Act and the Creative Commons license—CC BY-NC-ND (Attribution, Non-Commercial, No Derivative Works). Under this license, works must always be attributed to the copyright holder (original author), cannot be used for any commercial purposes, and may not be altered. Any other use would require the permission of the copyright holder. Students may inquire about withdrawing their dissertation and/or thesis from this database. For additional inquiries, please contact the repository administrator via email (scholarship@uwindsor.ca) or by telephone at 519-253-3000ext. 3208.
An Approach of QoS Evaluation for Web Services Design with Optimized Avoidance of SLA Violations

By

Peng Cheng

A Dissertation
Submitted to the Faculty of Graduate Studies through the School of Computer Science in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy at the University of Windsor

Windsor, Ontario, Canada

2020

©2020 Peng Cheng
An Approach of QoS Evaluation for Web Services Design with Optimized Avoidance of SLA Violations

by

Peng Cheng

APPROVED BY:

__________________________
H. Lutfiyya, External Examiner
University of Western Ontario

__________________________
H. Wu
Department of Electrical and Computer Engineering

__________________________
A. Ngom
School of Computer Science

__________________________
D. Wu
School of Computer Science

__________________________
X. Yuan, Advisor
School of Computer Science

February 14, 2020
Declaration of Co-Authorship/Previous Publication

I thereby certify that this dissertation incorporates material that is the result of my research conducted under the supervision of my advisor Dr Xiaobu Yuan. Parts of Chapter 3 and 4 of the dissertation was co-authored with Dr Xiaobu Yuan. In all cases, the key ideas, primary contributions, justifications of the methodologies, the design of the experiments, data analysis and interpretations, were performed by myself with the guidance of Dr Xiaobu Yuan.

I am aware of the University of Windsor Senate Policy on Authorship and I certify that I have properly acknowledged the contribution of other researchers to my thesis, and have obtained written permission from each of the co-author(s) to include the above material(s) in my thesis.

I certify that, with the above qualification, this thesis, and the research to which it refers, is the product of my own work.

This thesis includes two original papers that have been previously published / submitted for publication in peer reviewed journals, as follows:


I certify that I have obtained a written permission from the copyright owner(s) to include the above published material(s) in my thesis. I certify that the above
material describes work completed during my registration as a graduate student at the University of Windsor.

I declare that, to the best of my knowledge, my thesis does not infringe upon anyone’s copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.
Abstract

Quality of service (QoS) is an official agreement that governs the contractual commitments between service providers and consumers in respect to various non-functional requirements, such as performance, dependability, and security. While more Web services are available for the construction of software systems based upon service-oriented architecture (SOA), QoS has become a decisive factor for service consumers to choose from service providers who provide similar services. QoS is usually documented on a service-level agreement (SLA) to ensure the functionality and quality of services and to define monetary penalties in case of any violation of the written agreement. Consequently, service providers have a strong interest in keeping their commitments to avoid and reduce the situations that may cause SLA violations.

However, there is a noticeable shortage of tools that can be used by service providers to either quantitively evaluate QoS of their services for the predication of SLA violations or actively adjust their design for the avoidance of SLA violations with optimized service reconfigurations. Developed in this dissertation research is an innovative framework that tackles the problem of SLA violations in three separated yet connected phases. For a given SOA system under examination, the framework employs sensitivity analysis in the first phase to identify factors that are influential to system performance, and the impact of influential factors on QoS is then quantitatively measured with a metamodel-based analysis in the second phase. The results of analyses are then used in the third phase to search both globally and locally for optimal solutions via a controlled number of experiments. In addition to technical details, this dissertation includes experiment results to demonstrate that this new approach can help service providers not only predicting SLA violations but also avoiding the unnecessary increase of the operational cost during service optimization.
Dedication

This dissertation is dedicated to my wife, Juan Fan, and to my parents, Maidi Su and Xiaoxin Cheng, for their endless love, support, and encouragement.
Acknowledgements

It gives me immense pleasure to thank a few very important people who have helped me a lot during the whole process of my graduate studies.

First and foremost, I would like to present my gratitude to my supervisor Dr. Xiaobu Yuan for his valuable assistance and support during the past fours years. This work could not have been achieved without his continuous encouragement, valuable advices, suggestions and cooperation.

I would also like to thank my committee members, Dr. Huapeng Wu, Dr. Alioune Ngom and Dr. Dan Wu for their encouragement, insightful comments, and advice.

Finally, I want to thank my wife and parents for their love and support throughout my life.
Contents

Declaration of Co-Authorship/Previous Publication III

Abstract V

Dedication VI

Acknowledgements VII

List of Tables XI

List of Figures XIII

1 Introduction 1

1.1 Background .................................................... 1

1.2 Research Contributions ....................................... 5

1.3 Overview of the Unified Framework ............................ 6

1.4 Dissertation Organization ...................................... 8

2 Background and Literature Review 9

2.1 Service-Oriented Architecture (SOA) ......................... 9

2.2 Service Level Agreement (SLA) .............................. 12

2.3 SLA Violation Prediction ..................................... 16

2.4 QoS Optimization .............................................. 19

2.5 The Design and Analysis of Experiments .................... 24

2.5.1 Motivation: Why the Design and Analysis of Experiments ... 24

2.5.2 The Basic Concept Of the Design and Analysis of Experiments 25
2.5.3 Performance Analysis via the Design and Analysis of Experiments 28

2.5.4 Performance Optimization via the Design and Analysis of Experiments ........................................ 30

3 Sensitivity and Metamodel-Based Analyses for Services Design 32

3.1 Introduction ......................................................... 32

3.2 Metamodeling for Prediction of SLA Violation ................. 33

3.2.1 Multi-Factor Sensitivity Analysis ............................... 34

3.2.2 Metamodel-based Performance Analysis ........................ 37

3.3 Experiments ......................................................... 40

3.3.1 Experiments Setup ................................................. 40

3.3.2 Design and Perform Experiments ............................... 40

3.3.3 Perform Screen Design ........................................... 42

3.3.4 Specify One Appropriate Type of Metamodel .................. 43

3.3.5 Specify a Series of Candidate Forms for the Metamodel and
Construct These Surrogate Models ................................. 43

3.3.6 Check Model Adequacy ........................................... 45

3.3.7 Verify the Fitted Model ........................................... 47

3.3.8 Analyse the Effects of Factors and their Interaction on Response
Time .............................................................................. 48

3.3.9 Prediction Accuracy Comparison .................................. 51

3.4 Summary ............................................................... 55

4 A Strategy of QoS Optimization for Web Services 56

4.1 Introduction ........................................................... 56

4.2 Definition of the Optimization Problem and Related Conceptual Models 57

4.2.1 Optimization Context Model ..................................... 57
5 Conclusions and Future Works

5.1 Conclusions

5.2 Future Works

Bibliography

Vita Auctoris
List of Tables

2.1 Risk Assessment-based Methods ........................................ 14
2.2 SLA Management-based Methods ....................................... 15
2.3 Collaborative Filtering (CF)-based Methods ......................... 17
2.4 Methods of SLA Violation Prediction .................................. 20
2.5 QoS Optimization Methods Based on Service Composition ........ 21
2.6 QoS Optimization Methods Based on Resource Provisioning ..... 22
3.1 Data Collection ............................................................. 41
3.2 Variance Table .............................................................. 42
3.3 Data Collection ............................................................. 43
3.4 $R^2, R^2_{adj}$ and $R^2_{pdl}$ of fitted models ......................... 44
3.5 Results of Verifying Experiments ...................................... 49
3.6 Comparison Based on Different Number of Factors ............... 53
3.7 Comparison Based on Different Models ............................... 54
4.1 M-Factor Variance Table .................................................. 63
4.2 Result of Experiment for Global Search Algorithm ................ 73
4.3 The Fitted Mars Model for the Benchmark Function ............... 74
4.4 Subregions and Their Starting Search Points ....................... 75
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.5</td>
<td>Result of Experiments on Local Search Algorithm</td>
<td>76</td>
</tr>
<tr>
<td>4.6</td>
<td>Comparison of Approximation and Prediction Accuracy of Fitted Global Models</td>
<td>78</td>
</tr>
<tr>
<td>4.7</td>
<td>Optimization Results by Using Response Surface Methodology</td>
<td>80</td>
</tr>
<tr>
<td>4.8</td>
<td>Optimization Results by Using Global Metamodel-based Optimization Methodology</td>
<td>81</td>
</tr>
<tr>
<td>4.9</td>
<td>Robustness Comparison of Approximation Models</td>
<td>82</td>
</tr>
<tr>
<td>4.10</td>
<td>Details of the Four Additional Functions</td>
<td>82</td>
</tr>
<tr>
<td>4.11</td>
<td>Response Time of Web Service</td>
<td>85</td>
</tr>
<tr>
<td>4.12</td>
<td>Fitted Mars Model for the Web Service</td>
<td>85</td>
</tr>
<tr>
<td>4.13</td>
<td>Subregions and Starting Search Points</td>
<td>86</td>
</tr>
</tbody>
</table>

XII
List of Figures

1.1 Three Web Service Roles ........................................ 2
1.2 A Unified Framework of QoS Optimization ..................... 8
2.1 General model of a process or system ............................ 26
3.1 A Unified Framework of Sensitivity and Metamodel-Based Analyses . 34
3.2 Normal Plot Of Residuals. ........................................ 45
3.3 Externally Studentized Residuals. ............................... 46
3.4 Residuals VS. Run. ............................................. 47
3.5 Predicted VS. Actual. ............................................ 48
3.6 Three-dimensional Response Surface Plot of the fitted model. .... 50
3.7 Interaction between the Number of Sources and the Number of Products with Respect to Response Time. ...................... 51
3.8 Contour plot between the number of sources and the number of products with respect to response time. ......................... 52
4.1 A Unified Framework of QoS Optimization ........................ 62
4.2 The GRAMACY-LEE (2012) function. ............................. 72
4.3 The Comparison between All Fitted Global Models ............... 78
4.4 The Comparison Between the True Model and the Fitted Model ... 79
Chapter 1

Introduction

1.1 Background

The dramatically increased software complexity has made productivity and time-to-market two of the major concerns of software industry. Traditional approaches of software development failed to cope with sophisticated computer applications. In comparison, Service-Oriented Architecture (SOA) has emerged as a new technology that helps organizations to deal with the interoperability of software systems in heterogeneous environments. An SOA system consists of a collection of services that interact with each other, and its inter-service infrastructure focuses on Web services when services communicate via the Internet. SOA promises added benefits of easier components reuse, increased reliability, and reduced development and deployment costs [46].

More recently, Web services have become the technology of choice for realizing Service-Oriented Architecture and its associated set of strategic goals [1]. Web services are considered as self-contained, self-descriptive, modular applications that can be published, located, and invoked across the Web [2]. The communication between Web services is done via standards-based technologies that give users the opportunity
to access different Web services, independent of their hardware, operating system, or even programming environment. This supports organizations with a technology to create services that can be easily discovered and consumed by external users [1]. Software can be redelivered and paid for as streams of services as opposed to packaged products, thus making it possible to achieve automatic and dynamic interoperability between systems while accomplishing business tasks [6]. Businesses can be released from the burden of complex, low, and expensive software integration and focus instead on the value of their mission and critical tasks. Consequently, the Internet becomes a global common platform where organizations and individuals communicate with each other to carry out various commercial activities and to provide value-added services [6]. The rest of this section briefly presents the main concepts, principles and research problems relevant to Service-Oriented Architecture. In Chapter 2, the background and literature review of Service-Oriented Architecture will be discussed in more details.

As shown in Fig. 1.1, Web services play three major roles in an SOA system. With service registry acting as a centralized directory, service providers publish the information of their Web services for service consumers to select according to their needs of software development. The selected Web services are then bound at run time and invoked during the operation of software systems. Usually, Web services are third-party software made available by service providers, which means that these distributed applications based on Web services rely on these providers to ensure that their services comply with the agreed Quality of Service (QoS) [3]. Meanwhile, with tens of thousands of Web services available on the Internet, many of which are of equiva-

![Figure 1.1: Three Web Service Roles](image-url)
lent functions, QoS values are important criteria for service consumers to choose the web services that meet their requirements [5]. Many investigations have been carried out focusing on different kinds of QoS-related issues, such as Web service selection, Web service composition, SOA failure prediction, SOA performance prediction, fault tolerance, resiliency quantification, service searching, resource consistency, resource allocation, workload balance, as well as dynamically resource management [10]. As a result, QoS is currently a leading research topic in the field of SOA [3].

QoS is widely employed for describing non-functional characteristics of Web services. With the increasing number of Web services, QoS has become an important differentiating point of different functionally equivalent Web services. A Web service’s QoS includes a number of properties, such as response time, throughput, failure probability, availability, price, popularity, and so on [4]. As a service-level agreement (SLA), QoS is documented to guarantee that services fulfill their official commitments in terms of both functionality and quality. While service consumers focus on suitability when selecting from a pool of similar candidates, service providers compete with each other to offer not only the anticipated functionality but also attractive quality. If service provider fails to ensure SLA for service consumers, this will have negative impact on customer satisfaction and even lead to a disaster. For instance, when Amazon Elastic Cloud crashed in 2011 in terms of availability, it faced an outage and many big customers such as Quora and Reddit were down for more than a day. Such crashes affects service providers and service consumers. As SLA also defines monetary penalties in case of any violation of the written agreement, service providers have a strong interest in keeping their commitments to avoid and reduce the situations that may cause SLA violations.

Nevertheless, avoidance of SLA violations has been proven to be a challenging task as QoS is influenced by a variety of factors [29]. Because of variations in workload, computing resources, and even network conditions, it is common for Web services to exhibit fluctuations in performance, leading to the possibility of violating SLA. Any service provider who does not take action to prevent SLA violation will have to face
monetary penalties and lost revenue due to damaged relationships with clients [38]. Therefore, service providers are in desperate need of such tools that can not only help them predict if their service design provokes SLA violations but also guide them to optimize service design and prevent SLA violations from happening. When SLA violations are predicted, service providers can always optimize QoS of Web services by configuring different contextual factors, such as CPU, storage, distributed computing, or data cache though at the risk of increased operational cost due to over provision [38].

The extraordinary potential of SOA in software development has stimulated continuous research in the academic community and explorations in the software industry, as to be discussed in the next chapter of the literature survey. To improve the design of Web services, encouraging progress has been achieved for the evaluation of QoS, prediction of SLA violations, and QoS optimization. For example, most of the approaches use the technique of collaborative filtering to evaluate SOA systems with historical datasets obtained from other systems running a similar application and to predict the possibility of SLA violations based upon the calculation of QoS values for existing services [12, 13, 15, 16, 17, 74]. For QoS optimization, some approaches work for groups of services by maximizing aggregated utility values [89, 90, 91, 92] and others for individual services by allocating more resources [87, 88]. However, current methods still lack objective evaluation of software operating in different application domains, and their separation of related processes has resulted in limited success in practice.

In this dissertation, the avoidance of SLA violation is divided into two subproblems: QoS evaluation and optimization. For QoS evaluation, the dissertation research first applies the principle of design of experiments to analyze how sensitive a system is influenced by uncontrollable factors of a Web service and to find the influential factors that dominate the service's performance. The identification of influential factors then helps to develop an algorithm that constructs a surrogate model for the Web service and to use it to predict situations in which violations of SLA may take place. QoS
optimization is necessary only when SLA violations are supposed to happen, and it is carried out in the dissertation research by a guided search to globally identify sub-regions with local optima and to locally locate the best optimum with constraints to minimize operational cost. In such a way, the three normally separated processes of sensitivity analysis, SLA violations predication, and QoS optimization are integrated into a unified framework to help service providers to analyze, predict, and prevent SLA violations.

1.2 Research Contributions

This dissertation makes the following contributions in QoS evaluation for Web services design with optimized avoidance of SLA violations:

1. Multi-factor sensitivity analysis:
   
   (a) Created a general multi-factor model of sensitivity analysis that decomposes the total sensitivity measure into several sensitivity measures for each of individual factors and for all their joint effects in different combinations.
   
   (b) Based upon this general model, developed a new algorithm of sensitivity analysis for the identification of influential factors. As the algorithm calculates the influence of different input parameters on the model’s output and identifies the most influential parameters, it can be used to improve model quality and to reduce parametrization and computational costs. In addition, this algorithm requires a relatively small number of experiments with high accuracy and robustness.

2. Metamodel-based performance analysis:

   (a) The identification of influential factors helps to overcome the problem of over-parameterization and allows the development of a novel algorithm to create a fitted metamodel for quantitative analysis of QoS.
(b) This algorithm of performance analysis uses the metamodel for quantitative analysis of service performance and for the prediction of SLA violations without domain limits.

3. QoS optimization:

(a) Created a formal model with influential controllable factors for the problem of QoS optimization.

(b) Developed a search algorithm to fit a global surrogate model with multivariate adaptive regression splines and to identify sub-regions with potential local optima.

(c) Extended the response surface methodology to allow the search algorithm finding the global optima among the local optimum in the identified sub-regions. Furthermore, the algorithm takes minimizing operational costs into consideration to prevent over-provisioning resources.

1.3 Overview of the Unified Framework

Sensitivity analysis, SLA violation prediction, and design optimization are three important yet normally separated techniques for QoS evaluation and optimization [67]. As illustrated in Fig. 1.2, this dissertation research proposes an innovative framework to tackle the main issues of SLA violation by joining these three techniques into a unified framework to analyze, predict, and prevent SLA violations. Based upon a newly invented multi-factor model, the process of sensitivity analysis in this framework initiates the construction of a variance table with a chosen experimental strategy, e.g. Taguchi’s orthogonal arrays [68], by calculating the F-distribution values to determine the impact of factors, both individually and jointly, that are beyond the control of a service on a Web service’s performance. Among all the uncontrollable factors, those whose individual or joint F-distribution values exceed the threshold value of a cumulative F-distribution are influential factors of the service.
With the number of uncontrollable factors being reduced to only those influential to the service's performance, the second process treats the Web service under examination as a black box, use these identified influential uncontrollable factors as input variables and fits it with a surrogate model, or metamodel. The development of a metamodel starts with a candidate in a specific type and form, such as a quadratic model in polynomial regression, from a pool of choices. The process of SLA violation prediction then checks the adequacy of this chosen candidate with new experiments and verifies that the observable performance merits of the metamodel are within the 90% prediction interval at a residual error rate less than 10%. Unless a rejection results in the choice of another candidate, the metamodel is ready to be used for quantitative analysis of service quality and for the prediction of situations when SLA violations could take place. Meanwhile, the metamodel can help service provider better understand the performance of a service and identify performance bottlenecks caused by influential factors, which can help service provider choose the effective controllable factors to optimize performance in the next phase.

The last process of QoS optimization becomes inevitable to avoid the predicted SLA violations by adjusting the values of controllable factors by service providers, such as CPU, storage, and cache. This process includes five steps as below:

1. Identify the most influential controllable factor;
2. Conduct experiments and collect observation data;
3. Fit the collected dataset with the model of multivariate adaptive regression splines (MARS);
4. Search globally and create a list of subregions in which potential local optima exist;
5. Identify global optimum by searching local optima.

This five-step procedure repeats until either SLA prevention is achieved or the
1.3 Sensitivity Analysis

Construct Variance Table
Evaluate F-Values
Find Influential Factors

Sensitivity Analysis

Predict SLA Violation

Fit with A Metamodel
Analyze Model Adequacy

Evaluate F-Values

SLA Violation Prediction

Check Influential List
Identify Global Optimum
Search Local Optimum
Obtain List of Subregions

Experiment with A Factor

QoS Optimization

exhausted
no
SLA assured

completed
no

Figure 1.2: A Unified Framework of QoS Optimization

list of controllable factors is exhausted. In the latter case, a redesign of the Web service by the provider becomes a sensible recommendation.

1.4 Dissertation Organization

The rest of this dissertation is organized as follows. Chapter 2 introduces the background of SOA and the latest literature reviews related to SLA violation, SLA violation prediction, QoS optimization and the design and analysis of experiments. After presenting the first and second phases of the proposed framework on SLA violation prediction in Chapter 3, Chapter 4 presents the third phase of the proposed framework on QoS optimization for Web services. Finally, conclusions with some suggestions for future works are given in Chapter 5.
Chapter 2

Background and Literature Review

2.1 Service-Oriented Architecture (SOA)

The dramatically increased complexity of software systems in the past decades has made traditional approaches of software development inadequate for the software industry to handle sophisticated computer applications. In the demand of new technologies, SOA has emerged as an efficient way to help organizations dealing with the interoperability of software systems in heterogeneous environments. SOA offers the much-needed benefits of easier components reuse, increased productivity, improved reliability, shorter time-to-market, and reduced deployment costs. It is one of the most successful architectural styles in which applications make use of reusable services via internet. In the next decade, the SOA principles will be at the core of a new era of business engagements that transact at Internet scale across locations, devices, people, processes and information [1].

An SOA system consists of a collection of services that interact with each other, and its inter-service infrastructure focuses on Web services when services communicate via the Internet. It is based on some key principles listed as follows [69]:

[1]
1. Loose coupling: Services maintain a relationship that minimizes dependencies and only maintain an awareness of each other. This principle advocates the creation of a specific type of relationship within and outside of service boundaries, with a constant emphasis on reducing loosening dependencies between the service contract, its implementation, and its service consumers.

2. Service contract: Services adhere to a communicative agreement as defined collectively by one or more service description documents. Services express their purpose and capabilities via a service contract.

3. Service abstraction: Beyond what is described in the service contract, services hide logic from the outside world. Abstraction ties into many aspects of service-orientation. On a fundamental level, this principle emphasizes the need to hide as much of the underlying details of a service as possible.

4. Service reusability: Services provide specific functionality with the intention of promoting reuse. Reuse is strongly advocated within service-orientation; so much so, that it becomes a core part of typical service analysis and design processes, and also forms the basis for key service models. The advent of mature, non-proprietary service technology has provided the opportunity to maximize the reuse potential of multi-purpose logic on an unprecedented level.

5. Service composability: Collections of services can be coordinated and assembled to form composite services. As the sophistication of service-oriented solutions continues to grow, so does the complexity of underlying service composition configurations.

6. Service autonomy: Services have control over the logic they encapsulate. For services to carry out their capabilities consistently and reliably, their underlying solution logic needs to have a significant degree of control over its environment and resources.

7. Service statelessness: The management of excessive state information can compromise the availability of a service and undermine its scalability potential.
Services are therefore ideally designed to remain stateful only when required.

8. Service discoverability: Services are designed to be outwardly descriptive so that they can be found and assessed via available discovery mechanisms.

Each service can be provided by different providers and used by one or multiple consumers. A service consumer can be an application or a service that reuses other services. According to the SOA systems that service consumers are developing, they choose the most suitable services from different candidates with similar functionality and use them to compose their application. A service provider, on the other hand, can be an individual or an organization that develops and maintains reusable services. These services are available for service consumers to reuse. Competition is extremely fierce between different service providers as there are always others with the same features. If a service is not capable of satisfying service consumers in terms of quality and functionality, service consumers may abandon this service and choose another service provider.

Web services offer a potential solution for developing distributed business processes and applications that can be accessed via the Internet. The use of Web services in SOA systems have have many benefits for the development of new applications [76]. While QoS encompasses various non-functional issues, such as performance, dependability, and security, SLA as an official document of QoS is in place to assure service consumers that they get the services paid for by obligating service providers to fulfill contractual promises. As more competitive Web services become available for service consumers to choose, QoS has become a decisive factor to distinguish the reputation of service providers [77]. In the next section, QoS and SLA will be discussed in detail.
2.2 Service Level Agreement (SLA)

As QoS tells the ability of a service to meet certain requirements for different aspects of the service, such as performance, availability, reliability, and cost, it plays an important role in the selection and composition of services for the development of SOA systems. For qualitative and quantitative QoS evaluation, service providers and customers negotiate the QoS attributes they mutually agree on, including rejection rate, mean time between failures, response time, throughput, financial cost, or energy consumption. As part of the contract between service consumers and providers that defines the level of service, all SLAs need to formally describe both the different Service Level Objectives (SLO) in terms of values for QoS metrics and penalties if the objectives have not been accomplished [70]. For instance, the following specifications could be in an SLA for a Web service that generates a list of top ten webpages linking to different e-commerce websites with the lowest prices for certain products.

1. The average response time for the operation of this Web service should be less than 50 milliseconds.

2. This service must be available from 6:00 AM to 8:00 PM, Monday through Sunday.

3. The service provider should seek 100% availability during working hours for this Web service.

4. A penalty may be applied upon the service provider if the fault lies on the side of the service provider. The penalty can also be waived if both parties reach an agreement to do so. If applied, the penalty will be: Penalty = Amount * Times. In this equation, the penalty is the total compensation service consumer will receive from the service provider for SLA violation in one year; Amount is the penalty for each SLA violation; Times is the total number of SLA violations taking place in one year.
SLA helps service consumers and providers understand the trade-offs inherent among cost, schedule, and quality with explicitly documented contracts, but the fulfillment of contracts has proven to be difficult. A report published by Forrester Consulting indicates that service providers fail to meet the expected outcomes for service consumers 26% of the time, directly resulting in 57% of the service providers noticing a decrease in revenue and negative impact on customer satisfaction due to their failure in meeting SLA requirements [71]. As a result, the avoidance of SLA violation for SOA has become an import topic of research that attracts interest from both the academic environment and the software industry.

By introducing a utility model for contract-based service provisioning, a method was developed in [34] to help service providers establish an optimal service provision contract under uncertainty. Based on the idea of risk assessment, this work focuses on SLA criteria formation and evaluation, instead of the quantitative measurement. Similarly, the authors of [84] proposed an optimal SLA formation architecture, enabling service providers to consider consumers’ reliability in their commitment to SLA. The authors classify existing consumers into three categories based on their reliability or trustworthiness value and use that knowledge to ascertain whether to accept a consumer request for resource allocation and then to determine the extent of the allocation. Their approach enables service providers to monitor the behavior of service consumers in the post-interaction time phase and to minimize service violations by using the information of consumer behavior to form viable SLAs in the pre-interaction time phase. From a risk management perspective, another paper proposed a process of SLA violation detection and suggested service providers to consider abatement of SLA violations with SLA monitoring, violation prediction, and recommendations for risk reduction [35].

Meanwhile, some SLA management-based methods have been proposed to prevent the violation of SLA. By introducing a mechanism for contract renegotiation and modification, a method was developed in [36] to revise the current format of SLA specifications and allows the involved parties of an SLA to renegotiate at run-time. It
Table 2.1: Risk Assessment-based Methods

<table>
<thead>
<tr>
<th>Publication</th>
<th>Contributions</th>
</tr>
</thead>
</table>
| Mitigating provider uncertainty in service provision contracts by C Smith and A Van Moorsel (2010) [34] | - Introduce a utility model for contract-based service provisioning.  
- Help service providers establish an optimal service provision contract under uncertainty. |
| Provider-based optimized personalized viable SLA (OPV-SLA) framework to prevent SLA violation by W Hussain, FK Hussain, OK Hussain and E Chang (2016) [84] | - Introduce a novel optimal SLA formation architecture.  
- Help the service provider monitor the behavior of service consumers in the post-interaction time phase. |
- Help the service provider to form an SLA which comprises SLA monitoring, violation prediction and decision recommendation. |

avoids both the suspension of service provisioning and the brutal termination of the contract. After examining how Service Level Agreement management (SLAM) deals with the dynamic nature of cloud and inspecting issues concerning SLAM, the authors of [37] suggested SLA renegotiation as an efficient way of managing SLA within a cloud-based system. To guarantee SLA for cloud services, a recent paper proposed a cloud model [70] that used a new language to formally describe cloud-oriented SLAs and to define different Service Level Objectives with utility functions. It developed several online controlling algorithms to monitor and ensure the SLOs for the objective of achieving a systematic and transparent integration of SLAs in the cloud.
Table 2.2: SLA Management-based Methods

<table>
<thead>
<tr>
<th>Publication</th>
<th>Contributions</th>
</tr>
</thead>
</table>
- Allow the involved parties of an SLA to renegotiate at run-time.  
- Avoid both the suspension of service provisioning and the brutal termination of the contract. |
| Renegotiation in service level agreement management for a cloud-based system by AFM Hani, IV Paputungan and MF Hassan (2015) [37] | - Propose the approach managing Service Level Agreement within a cloud-based system.  
- Use SLA renegotiation as an efficient way of managing SLA within a cloud-based system. |
| SLA guarantees for cloud services by D Serrano, S Bouchenak and Y Kouki (2016) [70] | - Formally describe cloud-oriented SLAs.  
- Proposed a model to guarantee SLA for cloud services.  
- Develop several online controlling algorithms to monitor and ensure the SLOs. |

Listed in Table 2.1 and Table 2.2 are two groups of publications surveyed in this section, with highlights of their contributions. While both risk assessment-based and SLA management-based methods can help service providers reduce or minimize SLA violations, existing approaches in these directions lack the capability of optimizing QoS according to SLA requirements when SLA violations are predicted or take place. As discussed in Section 1.1 on background, service providers who do not take action to optimize the performance when SLA violation occurred will not only face monetary penalties but also results in lost revenue due to their damaged relationship with service clients.
consumers [38]. Meanwhile, QoS is largely impacted by a variety of factors in the real-world service invocation. The typical number of controllable and uncontrollable factors that service providers need to deal with ranges from one to eight factors. Therefore, service providers need effective tools to help them analyze how different factors affect the QoS, to predict the situations in which these factors' influence cause SLA violations, and to optimize performance when an SLA violation is predicted or happens.

2.3 SLA Violation Prediction

In recent years, SLA violation prediction has become a popular topic in the study of cloud services. This subject can be viewed either from the perspective of the service consumer or from that of the service provider.

For SLA violation prediction from the perspective of the service consumer, most of the existing works focus on using or extending approaches based on collaborative filtering (CF), which encompasses techniques for matching people with similar interests and making recommendations on this basis. Generally, collaborative filtering algorithms can be divided into two main categories: memory-based and model-based algorithms. Memory-based methods employ similarity computation with past usage experiences to find similar users and services for making the performance prediction and, include user-based approaches, item-based approaches, and their fusion [14]. Memory-based CF algorithms are easy to implement and highly effective, but they suffer from a fundamental problem: the inability to scale-up. To address the scalability issue, model-based methods were proposed to utilize the models (such as clustering models, latent factor models, aspect models, and so on) to recognize complex patterns by seeking users for the recommendation within smaller and highly similar clusters, rather than within the entire database [29]. However, model-based methods are often time-consuming to build and update. Listed in Table 2.3 are publications surveyed
Table 2.3: Collaborative Filtering (CF)-based Methods

<table>
<thead>
<tr>
<th>Publication</th>
<th>Contributions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qos-aware Web service recommendation by collaborative filtering by Z Zheng, H Ma, MR Lyu and I King(2011) [12]</td>
<td>- Present a user-collaborative mechanism to predict QoS values of Web services.</td>
</tr>
</tbody>
</table>

in the direction of CF-based methods, with highlights of their contributions. After presenting a user-collaborative mechanism for past QoS information that are collected from different users, a collaborative filtering approach is proposed in [12] to predict QoS value of Web services and make Web service recommendation by taking advantages of past usage experiences of service users. To avoid the issue of the expensive and time-consuming invocation of Web services in [12], a neighborhood-integrated approach is designed in [13] for QoS prediction of personalized Web services. Based on the fact that the quality of Web service is affected by its context feature, the au-
thors of [16] proposes a new QoS-aware Web service recommendation system, which considers the contextual feature similarities of different services. To address the data credibility problem that most existing approaches ignore, the authors of [15] propose a trust-aware approach for reliable personalized QoS prediction. Meanwhile, to support timely and accurate adaptation decisions, the authors of [17] proposed an adaptive matrix factorization approach to perform online QoS prediction for candidate services. The approaches based on collaborative filtering operate on a common assumption that multiple applications are likely to support similar functionalities when the evaluation is conducted similarly for a large part of their services, and, therefore, their remaining services can be evaluated similarly [24]. However, when the new systems under development are for different application domains, their design involves significant structural or contextual changes and thus destroys the applicability of collaborative-filtering-based approaches. As a result, there is a demand for a more general approach that is capable of predicting the quality of services without the constraints on application domains.

For SLA violation prediction from the perspective of the service provider, several approaches have been proposed in recent years. Listed in Table 2.4 are publications surveyed in this direction, with highlights of their contributions. To guarantee the business requirements of service providers and maximize their profit, the authors of [83] proposed an intelligent and profile-based model of SLA violation prediction. This prediction model intelligently predicts the consumer’s likely resource usage by considering the consumer’s reputation from its previous transaction history and determines the level of required resources based on their reliability. While this method can help service providers make decisions about whether to form SLA and recommends how to mitigate those violations to avoid any penalties, it lacks the capability of predicting SLA violation in post-interaction time phase when the SLA contract has already been made between the service provider and the service consumer. To achieve service level agreements, a prediction method based on Bayes model was designed in [72] to predict the mean load over a long-term time interval, as well as the mean load in
consecutive future time intervals by identifying novel predictive features of host load that capture the expectation, predictability, trends and patterns of host load. This prediction model of the workload can help service provider estimate the possibility of whether SLA violation will occur. Meanwhile, the authors of [73] proposed to use the SVM model to predict possible SLA violation before any issue emerges so then remedial action can be taken. While the approaches in [72] and [73] can help the service provider know beforehand whether SLA violation will take place, it lacks the capability of helping service provider evaluate QoS quantitatively. However, service providers need an effective approach to measure and analyze the impact of influential factors on system performance. As a result, this dissertation will develop a novel approach to help service providers evaluate QoS by using techniques of the design and analysis of experiments. As part of the proposed unified framework, this approach uses sensitivity analysis to identify influential input factors of services, produces a fitted model to approximate the relationship between factors and QoS, and uses this fitted model to analyze and predict QoS values. In Chapter 3, this approach will be discussed in more details.

2.4 QoS Optimization

The subject of QoS optimization can be either composite or individual services. For composite services, several approaches that are listed in Table 2.5 suggested to calculate aggregated QoS values of all possible service combinations and choose the one that maximizes the aggregated utility value while satisfying global constraints [89, 90, 91, 92]. While most of these approaches in this direction are proved to be able to optimize QoS of the composite services effectively and efficiently, they become inapplicable for the individual service.

For individual services, resource provision has been utilized in [87] to analyze workloads, to categorize them on the basis of common patterns, and to plan for
### Table 2.4: Methods of SLA Violation Prediction

<table>
<thead>
<tr>
<th>Publication</th>
<th>Contributions</th>
</tr>
</thead>
</table>
| Profile-based viable service level agreement (SLA) violation prediction model in the cloud by W Hussain, FK Hussain and O Hussain (2015) [83] | - Introduce an efficient system that predicts SLA violation before it occurs and recommends how to mitigate those violations to avoid any penalties.  
- Propose a profile-based model of SLA violation prediction from the provider’s perspective.  
- Help service providers in making decisions about whether to form SLA and avoiding SLA violations. |
| Host load prediction in a Google compute cloud with a Bayesian model by Di Sheng, Derrick Kondo, and Walfredo Cirne (2012) [72] | - Use a Bayesian model to predict the mean workload.  
- Help service providers judge whether there is a possibility that SLA will be violated. |
| A machine learning model for detection and prediction of cloud quality of service violation by TS Wong, GY Chan and FF Chua (2018) [73] | - Use SVM model to help the cloud service provider know beforehand whether SLA violation will take place. |

workloads before actual scheduling. The authors of [87] later enhanced their work with automated processing in the [88]. Although these methods are able to optimize QoS by allocating more resources, no attention has been given to resource over-provision, which is a serious issue for Web services as it wastes resources and causes the increase of operational cost [38].

In this dissertation, optimization techniques are used to optimize QoS. Good optimization algorithms should possess the properties [97] listed below and should be taken into consideration when designing approaches for QoS optimization.
Table 2.5: QoS Optimization Methods Based on Service Composition

<table>
<thead>
<tr>
<th>Publication</th>
<th>Contributions</th>
</tr>
</thead>
</table>
| Optimal service selection and composition or service-oriented manufacturing network by S Huang, S Zeng, Y Fan and GQ Huang(2011) [89] | - Propose a performance evaluation model to analyse the local and global performance.  
- Realise the optimal service selection and composition. |
- Improves the reliability of dynamic Web service composition. |

1. Robustness: They should perform well on a wide variety of problems in their class, for all reasonable values of the starting point.

2. Efficiency: They should not require excessive computer time or storage.

3. Accuracy: They should be able to identify a solution with precision, without being overly sensitive to errors in the data or to the arithmetic rounding errors that occur when the algorithm is implemented on a computer.

Various types of optimization techniques are used to solve research problems in different fields. These approaches can be broadly classified into two strategies: direct
Table 2.6: QoS Optimization Methods Based on Resource Provisioning

<table>
<thead>
<tr>
<th>Publication</th>
<th>Contributions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q-aware: Quality of service based cloud resource provisioning by S Singh and I Chana (2015) [87]</td>
<td>- Cloud workloads have been analyzed and clustered through workload patterns.</td>
</tr>
<tr>
<td></td>
<td>- Identify QoS metrics of each workload.</td>
</tr>
<tr>
<td></td>
<td>- Analyze the effect of number of workloads and resources on execution time</td>
</tr>
<tr>
<td></td>
<td>and cost.</td>
</tr>
<tr>
<td></td>
<td>- Proposed technique for the minimization of cost and time simultaneously</td>
</tr>
<tr>
<td></td>
<td>while adhering to workload deadline.</td>
</tr>
<tr>
<td>STAR: SLA-aware autonomic management of cloud resources by S Singh, I Chana and R Buyya (2017) [88]</td>
<td>- Develop a resource management technique that automatically manages QoS requirements of cloud users.</td>
</tr>
<tr>
<td></td>
<td>- Help the cloud providers in achieving the SLAs and avoiding SLA violations.</td>
</tr>
</tbody>
</table>

and metamodel-based optimization.

For direct optimization, it is performed by evaluating real objective function. For the Service-Oriented system, the real objective function between influential factors and the performance is usually unknown, which means that the real performance model is regarded as a black box. Each evaluation of the real objective function typically involves the following steps: perform one experiment by configuring values of contextual factors, execute the designed Web services and collect the experimental data. As a result, the evaluation of real objective function for the Service-Oriented system is computationally expensive and involves cost. Meanwhile, when the formula of the real objective function is unknown, the heuristic optimization methods, such as genetic algorithms [98] and simulated annealing [99], are usually used to search the
optimum. These methods have fewer restrictions on the mathematical characteristics of the function, and do not require any gradient information. However, many objective function evaluations are usually needed for heuristic optimization methods, which means that a huge amount of time will be spent on performing these experiments. As a result, the strategy of direct optimization is inapplicable when optimizing the performance of the Service-Oriented system.

For Metamodel-based optimization, it makes use of some surrogate models to provide an efficient representation of the costly true model. When a surrogate model is used for this purpose, it is called a metamodel. This process is called metamodel-based design optimization (MBDO) [94, 95, 96] when metamodels are utilized for the evaluations during the optimization process. Metamodel-based design optimization usually includes three steps as below:

1. Perform a small or modest number of experiments and collect the experimental data;

2. Use these experimental data to fit a set of mathematical surrogate models that are used to approximate the real unknown function;

3. Identify the optimum by searching on the most appropriate surrogate model.

Since these surrogate models are mathematical functions with a specific formula, evaluation on these functions are efficient. As a result, the strategy of metamodel-based optimization will be used to optimize the performance of the Service-Oriented system in this dissertation. Because metamodel-based design optimization originates from techniques of the design and analysis of experiments, the relevant literature review will be discussed in the next subsection: design and analysis of experiments.
2.5 The Design and Analysis of Experiments

2.5.1 Motivation: Why the Design and Analysis of Experiments

In essence, QoS defined in the SLA is the quality attribute requirement for Web services. Whether a system will be able to exhibit its desired quality attributes is substantially determined by its architecture. In other words, quality attribute requirements are satisfied by the various structures designed into the architecture, and the behaviors and interactions of the elements that populate those structures [78]. As a result, a well-designed architecture can help service providers ensure SLA when designing Web services.

When performing architectural design, the architecture should be evaluated for its ability to deliver the system’s important quality attributes. This should occur early in the life cycle, when it returns the most benefit, and repeated as appropriate, to ensure that changes to the architecture (or the environment for which it is intended) have not rendered the design obsolete [78]. Meanwhile, there are various architectural tactics that can be used to achieve the requirement of quality attributes, such as provisioning more resources, caching and maintaining multiple copies of computation [79][80]. Service designers need to compare alternative architectural tactics and, hence, make well-informed design decisions when making trade-offs between cost, quality, and performance [33]. For example, when doing performance-driven architectural design for Web services, service designers may make different design decisions based on different environments in which Web services execute. Service designers usually need to ask some questions about the contextual information of Web services.

1. Will SLA be violated in the future?
2. Which factors have the significant influence on the performance?
3. How is the performance impacted by uncontrollable factors?

4. What are the performance bottlenecks that are caused by variations of different uncontrollable factors?

5. Which controllable factors can be used to improve performance effectively?

6. Which architectural tactics related to controllable factors can be used to improve the performance?

To answer these questions, service designers need to gain insight into the performance model and then can make the right design decisions based on this knowledge. Otherwise, a long time will be spent and wasted on the implementation of architectural tactics that will only give us a small percent performance boost. However, most of the answers to these questions are usually unknown during the architectural design stage. As a result, this kind of decision-making process for designing architecture is empirical and need to make use of techniques of the design and analysis of experiments (DAAE) to investigate the performance model.

2.5.2 The Basic Concept Of the Design and Analysis of Experiments

In general, the design and analysis of experiments (DAOE) is to use statistical techniques to analyze the results from designed experiments in an iterative manner. Specifically, we may want to determine which input variables are responsible for the observed changes in the response, develop a model relating the response to the important input variables and to use this model for system analysis, system improvement or other decision-making [8]. Because the results and conclusions that can be drawn from the experiment depend to a large extent on how the data were collected, the experiment is needed to be well-designed via the design of experiments. Design of experiments (DOE) is a test or series of tests in which changes are made on purpose
to the input variables of a process or system so that the causes for changes in the output response can be observed and identified. After data are collected from the well-designed experiment, some statistical techniques will be used to analyze these results to gain insight into the unknown system, to decide how to optimize the system or to make some other decision [8].

The motivation of design and analysis can be traced back to Fisher’s work on data analysis for agriculture experiments [75]. Since then, over thousands of activities and publications on it have been conducted and reported to characterize a system or product, to optimizing a product or process, and to design a product. Its applications cover a wide range of disciplines, including biology, agriculture, medicine and pharmacy, electronic and mechanics, and even social and human management [19, 104, 26, 27, 28].

The most commonly used terms in the design and analysis of experiments include controllable input factors, uncontrollable input factors, responses, hypothesis testing, and interaction.

1. Controllable input factors are those input parameters that can be controlled or configured, such as CPU, storage, cache, etc.

2. Uncontrollable input factors are those parameters that can not be controlled,

Figure 2.1: General model of a process or system
such as workload, network environment or the requirements of services consumer.

3. Responses or output measures are the elements of the process outcome.

4. Hypothesis testing helps determine the significant factors using statistical methods.

5. An interaction is a situation in which the simultaneous influence of two variables on a third is not additive when an experiment has three or more variables.

In general, experiments are used to study the performance of processes and systems. The process or system can be represented by the model shown in Fig. 2.1. The process can be visualized as a combination of operations, machines, methods, people, and other resources that transforms some input into an output that has one or more observable response variables. Some of the process variables \([x_1, x_2, \ldots, x_n]\) are controllable, whereas other variables \([z_1, z_2, \ldots, z_n]\) are uncontrollable. The overall objectives of the experiment include[8]:

1. Understanding: two main objectives of the design and analysis of experiments is to learn which factors have the most influence on the response(s) of interest and to understand the behavior of a system. Specifically, in order to avoid wasting valuable resources, to reduce the total number of experiments and to get a surrogate model with better quality, the service designer may be interested in understanding whether some factors, such as workload, CPU, memory, cache, etc, influence the performance of a service and get a list of important factors that have significant impact on the performance. They can apply techniques of DOAE to do sensitivity analysis to screen factors. Meanwhile, the service designer is interested in understanding the performance model that reflects the cause-effects relationship between influential factors and the response time. After constructing one surrogate model approximating the true performance model by using data from the well-designed experiments, it can be explored to help
service designers better understand the performance of Web services and predict SLA violation.

2. Optimization: after the system has been better understood, the next objective is usually optimization. That is to find the settings or levels of the important factors that result in desirable values of the response. An optimization experiment is usually a follow-up after the screening experiment, surrogate model fitting and exploration on this model.

3. Confirmation: in a confirmation experiment, the experimenter is usually trying to verify that the system operates or behaves in a manner that is consistent with some theory or experience.

2.5.3 Performance Analysis via the Design and Analysis of Experiments

To investigate the unknown system, several sensitivity analysis methods relying upon DOE were proposed to perform the factor screen. Sensitivity in this context is a measure of the contribution of an independent variable to the total variance of the dependent data. They can help us find how different model input parameters influence the model output, what are the most influential parameters and how to evaluate such effects quantitatively [39]. Meanwhile, when working with new systems or technologies, valuable resources will not be wasted by focusing on important factors and ignore unimportant ones [8].

By deriving measures of global sensitivity from a set of local derivatives or elementary effects, and sampling on a grid throughout the parameter space, a method of sensitivity analysis was proposed in the [40] to identify the main effects. In this paper, the proposed experimental plans are composed of individually randomized one-factor-at-a-time designs and data analysis is based on the resulting random sample of observed elementary effects. While the method in the [40] can identify the main
effects, it becomes inapplicable in terms of interaction effects. Similarly, the authors of [41] later use metamodeling approaches to estimate sensitivity indices. Instead of evaluating real model or performing the real experiment, this method first perform several experiments, then fit a surrogate model and finally calculate the sensitivity indices by using the data coming from the evaluation on the fitted surrogate model. Although this method can reduce the number of the experiment, its accuracy depends on the surrogate model. If the fitted model is not accurate enough, the sensitivity indices will be not capable of providing useful information to make decision. To overcome the issues of lack of efficiency and accuracy, the authors of [60] and [21] present the investigation on performance analysis with sensitivity analysis and a generalization for the statistical method to handle two or three factors. The approach in these two papers can be used to analyze main and interaction effects, and require a relatively small number of experiments. However, the effects models and algorithms can only be used to analyze two or three factors. As a result, the research in this dissertation will enhance and extend the models and algorithms in [60] and [21] to improve its robustness.

In order to gain a deeper insight into the behavior of a system, many papers in a variety of fields use metamodeling techniques to further explore the unknown system and predict the system behavior. Meanwhile, Complex computational models often suffer from over-parameterization that can reduce model quality and increase computational costs [42]. As a result, before using metamodeling techniques to further explore the behavior of a system, sensitivity analysis can be used to exclude unimportant parameters to potentially improve model quality and to reduce parametrization. In recent years, many papers [19, 104, 26, 27, 28] covering a wide range of fields, including intelligent manufacturing, civil engineering, powder technology, electronic and mechanics, use techniques combing metamodel-based analysis with sensitivity analysis to evaluate the performance model. As a result, techniques of the design and analysis of experiments have already been proven to be an effective way to model the cause-effects relationship between influential factors and the performance, to explore
unknown performance system, and to optimize the performance. In this dissertation, sensitivity analysis combined with the metamodel technique will be employed to evaluate the performance model of Web services. In Chapter 3, this approach will be discussed in more details.

2.5.4 Performance Optimization via the Design and Analysis of Experiments

For optimization, many metamodel-based design optimization (MBDO) methods that rely upon DOE were proposed. Response surface methodology is the most popular metamodel-based optimization method used in recent years [100]. It has been studied and used for optimization in a variety of different fields [101, 102, 103, 104, 105] due to its efficiency and accuracy for finding the optimum. However, RSM uses the linear and quadratic model to approximate the real response model and can only guarantee to find a local optimum. When the real response model of the system is complex and has multiple local minimum or maximum optima, the global optimum can not be obtained in some cases. As a result, the global metamodel-based optimization approaches [106, 107, 108, 109, 110] are used to overcome this issue of being trapped by the local optimum and have a better chance of pinpointing the global optimum. These methods fit one or more global metamodels following by searching global optimum on those global models. When the global metamodel is used to approximate the true unknown model, it need to accurately capture the real model of the system to ensure the searching efficiency and accuracy. However, it is difficult to build a highly accurate global model because the error occurring in the global surrogate model will usually become larger than that in the local surrogate model when the global region becomes large. As a result, the global and local metamodel-based optimization approaches should be combined to take advantage of their merits. In this dissertation, QoS optimization is carried out by a guided search to globally identify subregions with local optima and to locally locate the best optimum with constraints
to minimize operational cost. In Chapter 4, this approach will be discussed in more
details.
Chapter 3

Sensitivity and Metamodel-Based Analyses for Services Design

3.1 Introduction

In support of software development based upon Service-Oriented Architecture, service providers use quality of service (QoS) to guide their design of services. By utilizing historical information of QoS collected from service consumers operating similar software systems, a group of approaches [12, 13, 15, 16, 17, 74] has recently been developed to analyze and predict QoS values for new systems with impressive accuracy as long as these systems fall into the same category of application domain, e.g., for buying, watching, or listening online [61]. Based upon collaborative filtering, those approaches rely on a common assumption that multiple software systems are likely to support similar functionality. After evaluating QoS for a large part of their services, the results are valid for the remaining services [62]. However, when the new systems under development are for different application domains, their design involves significant structural or contextual changes and thus destroys the applicability of collaborative-filtering-based approaches.
In the demand for a more general approach that is capable of predicting quality of services without the constraints on application domains, this chapter proposes a framework to use sensitivity analysis for the identification of influential factors with dominating impacts on the quality of services and use metamodel-based analysis to select a fitted surrogate model for the prediction of potential violation of SLA. In the remaining of this chapter, Section 3.2 provides details of sensitivity analysis and metamodel-based analysis after introducing a unified framework to integrate the two processes together. Afterwards, Section 3.3 goes through the main steps with experiment results on a real Web service and uses it as an example to illustrate the use of this framework for service design. Finally, this chapter concludes with discussions and future work in Section 3.4.

3.2 Metamodelling for Prediction of SLA Violation

Every service in an SOA system has a number of parameters as input variables, whose values can be either discrete or dividable into discrete segments. Assuming the true performance model of a service can be expressed as a function in (3.1),

$$ Y = f(A^{(1)}, A^{(2)}, \cdots, A^{(m)}) $$  \hspace{1cm} (3.1)

where $Y$ is the performance of service under the influence of $m$ input factors $A^{(v)}$, $1 \leq v \leq m$. As the true functional relationship between performance and factors is unknown, the prediction of SLA violation needs an approximation model $g(\cdot)$, or a metamodel, to imitate the behavior of $f(\cdot)$.

$$ \hat{Y} = g(A^{(1)}, A^{(2)}, \cdots, A^{(m)}) $$  \hspace{1cm} (3.2)

In practice, the metamodel will be different from the real model in most cases, i.e.,
3. SENSITIVITY AND METAMODEL-BASED ANALYSES FOR SERVICES DESIGN

Figure 3.1: A Unified Framework of Sensitivity and Metamodel-Based Analyses

\[ Y = \hat{Y} + \varepsilon, \text{ but } g(\cdot) \text{ is an adequately fitted model if the difference } \varepsilon \text{ is less than a predetermined threshold. Presented in this section is a unified framework (Fig. 3.1) that integrates the processes of sensitivity analysis for the identification of influential factors and metamodel-based analysis for the prediction of SLA violation.} \]

3.2.1 Multi-Factor Sensitivity Analysis

Sensitivity analysis is the study of how the uncertainty in the output of a system or mathematical model can be apportioned to different sources of uncertainty in its inputs [18]. As illustrated in (3.3) for a service with \( m \) input factors \( A^{(v)}, 1 \leq v \leq m \), the observation of an experiment with each factor taking a specific value indexed at \( i_v, 1 \leq i_v \leq a_v, \) is the system’s output \( Y_{i_1 \ldots i_m} \) collected with a performance metric, where \( 1 \leq l \leq n \) for \( n \) experiments.

\[
Y_{i_1 \ldots i_m} = \mu + Y_{i_1 \ldots i_m}' + Y_{i_1 \ldots i_m}'' + \varepsilon_{i_1 \ldots i_m} \quad (3.3)
\]
In (3.3), $\mu$ is the population mean obtained from the average of all $n$ observations per experiment arrangement, and $\varepsilon_{i_1...m}$ is a random error. In addition, the components of $Y'_{i_1...m}$ and $Y''_{i_1...m}$ counter in the influence of all factors individually and jointly as formulated in (3.4) and (3.5) respectively.

$$Y'_{i_1...m} = \sum_{v=1}^{m} A^{(v)}_{i_v}$$ (3.4)

$$Y''_{i_1...m} = \sum_{u=1}^{m-1} \sum_{v=1}^{m-w} \prod_{v=u}^{v=u+w} A^{(v)}_{i_v} \cdot \sum_{v=u+w+1}^{m} A^{(v)}_{i_v}$$ (3.5)

The preparation for sensitivity analysis needs a total number of $n \times \prod_{v=1}^{m} a_v$ experiments, from each of which an observed response $y_{i_1...m}$ is collected. The overall process of analysis goes through five steps to calculate individual/joint means, sum of squares, mean squares, and F distribution values for the identification of influential factors. In the first step, the overall mean $\hat{y}_{i_1...m}$ is obtained from (3.8), and all the other individual and joint means are obtained by simply taking out corresponding items from the nested sums in the dividend and from the product in the divisor of (3.8). For example, (3.6) tells how to calculate individual mean $\hat{y}_{i_2}$ for factor $A^{(2)}$ and (3.7) to calculate the joint mean $\hat{y}_{i_{23}}$ between factors $A^{(2)}$ and $A^{(3)}$.

$$\hat{y}_{i_2} = \frac{\sum_{i_1=1}^{a_1} \sum_{i_2=1}^{a_2} \cdots \sum_{i_m=1}^{a_m} \sum_{l=1}^{n} y_{i_1...m}}{a_1 a_2 \cdots a_m n}$$ (3.6)

$$\hat{y}_{i_{23}} = \frac{\sum_{i_1=1}^{a_1} \sum_{i_4=1}^{a_4} \cdots \sum_{i_m=1}^{a_m} \sum_{l=1}^{n} y_{i_1...m}}{a_1 a_4 \cdots a_m n}$$ (3.7)

$$y_{i_1...m} = \frac{\sum_{i_1=1}^{a_1} \sum_{i_2=1}^{a_2} \cdots \sum_{i_m=1}^{a_m} \sum_{l=1}^{n} y_{i_1...m}}{n \prod_{i=1}^{m} a_i}$$ (3.8)

Let $\hat{y}_I^2$ denote the square of overall mean $y_{i_1...m}$ and $\sum^{ai}$ symbolize the nested sums $\sum_{i_1=1}^{a_1} \sum_{i_2=1}^{a_2} \cdots \sum_{i_m=1}^{a_m}$. The second step of calculation produces the sum of squares $SS_{i_v}$ for each individual factor $A^{(v)}$ with (3.9), the sum of squares $SS_{i_1...m}$ for all $m$ factors with (3.10), and the error of sums of squares with (3.13), where $SS_J$ is specified in (3.11).
3. SENSITIVITY AND METAMODEL-BASED ANALYSES FOR SERVICES DESIGN

\[ SS_{i_v} = \frac{a_v}{n \prod_{i=1}^{m} a_i} \sum_{i_v=1}^{a_v} \hat{y}_{i_v} - \hat{y}_I^2 \]  
(3.9)

\[ SS_{i_1\ldots i_m} = \frac{1}{n} \sum_{a_i} a_i \sum_{i_1\ldots i_m} y_{i_1\ldots i_m}^2 - \hat{y}_I^2 - \sum_{i=1}^{m} SS_{i_i} - SS_J \]  
(3.10)

\[ SS_J = \sum_{u=1}^{m-1} \sum_{u+w} SS_{i_u} \sum_{i_v} \prod_{i=1}^{m} (a_i - 1) \]  
(3.11)

\[ SS_T = \sum_{a_i} a_i \sum_{i=1}^{m} y_{i_1\ldots i_m}^2 - \hat{y}_I^2 - SS_J \]  
(3.12)

\[ SS_E = SS_T - \sum_{i=1}^{m} SS_{i_i} - SS_J \]  
(3.13)

In addition, after converting \( \frac{1}{n} \) to \( \prod_{i=1}^{m} \frac{a_i}{a_i} \), the sum of squares for any other combinations of factors are obtained by keeping \( \hat{y}_I^2 \) and those items related to the factors in the numerator \( \prod_{i=1}^{m} a_i \) and all sum components in (3.10). For example, (3.14) illustrates the calculation of \( SS_{i_23} \) for two factors \( A^{(2)} \) and \( A^{(3)} \).

\[ SS_{i_23} = \frac{a_2 a_3}{n \prod_{i=1}^{m} a_i} \sum_{i_2=1}^{a_2} \sum_{i_3=1}^{a_3} \sum_{i_v} \prod_{i=1}^{m} \hat{y}_{i_23}^2 - \hat{y}_I^2 - SS_{i_2} - SS_{i_3} \]  
(3.14)

The third step of sensitivity analysis uses the results from the previous step to produce mean squares with (3.15) for the whole list of factors or any individual or joint factors by using the corresponding sum of squares as the dividend and keeping only related items in the product of the divisor. As illustrated in (3.16) for \( FA^{(1\ldots m)} \), the calculation of F distributions in the fourth step is a simple division of a mean square with \( MS_E \), which is the mean square error given by (3.17).

\[ MS_{A^{(1\ldots m)}} = \frac{SS_{i_1\ldots i_m}}{\prod_{i=1}^{m} (a_i - 1)} \]  
(3.15)

\[ FA^{(1\ldots m)} = \frac{MS_{A^{(1\ldots m)}}}{MS_E} \]  
(3.16)

\[ MS_E = \frac{SS_E}{(n - 1) \prod_{i=1}^{m} a_i} \]  
(3.17)
Finally, the process of sensitivity analysis compares each F distribution value with the cumulative F distribution value \( F_{\alpha, df_1, df_2} \), where \( \alpha \) is a confidence level, \( df_1 \) is the degree of freedom associated with the numerator of the mean square, and \( df_2 \) is the degree of freedom associated with the denominator the mean square [52]. A significant impact on the performance of a service can be identified if the F distribution value of an individual factor or a joint group of several factors exceeds \( F_{\alpha, df_1, df_2} \). By the end, this process produces a list of influential factors, each of which has significant impact on service performance individually or jointly with other factors.

### 3.2.2 Metamodel-based Performance Analysis

In the process of sensitivity analysis, the performance model \( f(\cdot) \) of a service is treated as a blackbox as influential factors are identified by analyzing only the outputs of selective inputs. As the prediction of SLA violation is expected to work in all use cases of the service, the proposed framework fits the approximation model \( g(\cdot) \) with a surrogate model and predicts service performance by evaluating this fitted model. While the elimination of insignificant factors in the previous process of sensitivity analysis greatly reduces the computational costs caused by over-parameterization of metamodeling techniques, the process of metamodel-based analysis starts with the identified influential factors and produces a surrogate model for the prediction of SLA violation. As illustrated in Fig. 3.1, the overall process goes through the following steps.

**Choose the type of metamodel**

This step requires experiments to be conducted with the influential factors using a chosen experimental strategy, which could be the Taguchi’s orthogonal arrays or any of the others as suggested in [7][8][115]. By inspecting the relationship between input factors and service performance after plotting experiment results onto a graph,
this step uses the distribution of marks in the graph to select a type of metamodel among the choices of polynomial regression, multivariate adaptive regression splines, support vector regression, or regression tree.

**Specify the form of the metamodel**

As a metamodel is not complete without specifying a form, the next step determines a selection of form from different candidates for the selected metamodel type. While linear model, two-factor interaction (2FI) model, and quadratic model are commonly used in polynomial regression, polynomial kernel and radial basis function kernel should be used for support vector regression.

**Check the adequacy of metamodel**

The objective of this step is to check the fitting quality of chosen metamodel in the selected form with experiments. If the candidate model lacks the required quality, the process of metamodel-based analysis repeats the work in the previous two steps to choose another metamodel. Otherwise, this step determines the most appropriate form of the chosen metamodel by evaluating its quality using a polynomial model defined by its coefficient of determination $R^2$, adjusted coefficient of determination $R_{adj}^2$, and predicted coefficient of determination $R_{pdt}^2$ as formulated below,

\[
R^2 = 1 - \frac{SS_E}{SS_T} \quad (3.18)
\]
\[
R_{adj}^2 = 1 - R^2 \times \frac{DOF_{SS_T}}{DOF_{SS_E}} \quad (3.19)
\]
\[
R_{pdt}^2 = 1 - \frac{SS_{PE}}{SS_T} \quad (3.20)
\]

where $SS_{PE}$ is prediction error sum of squares, and $SS_E$ and $SS_T$ are the error sum of squares and total sum of squares defined in (3.13) and (3.12) respectively, with their
corresponding degree of freedom given by $DOF_{SS_E}$ and $DOF_{SS_T}$. The fitted model is adequate only if the values of the three coefficients are within the range between 0.9 and the maximum of 1.0. Among them, $R^2$ and $R_{adj}^2$ are used to determine how well the model fits the experiment observations, and $R_{pdt}^2$ tells how well the model predicts responses for new observations. The larger value $R_{pdt}^2$ has, the greater predictive capability the fitted model exhibits.

**Verify the metamodel with new experiments**

After the analysis finishes validating the metamodel, it continues to assess the model’s accuracy when being used to predict service performance in different use cases. This assessment leads to a new series of experiments. Let a new observation be $x_0 = [1, x_{01}, x_{02}, \cdots, x_{0n}]$. A point estimate $\hat{y}_0$ for the future observation $y_0$ at the point $(x_{01}, x_{02}, \cdots, x_{0n})$ is calculated with the fitted model. The following formula gives the $(1 - \alpha) \times 100\%$ prediction interval for $y_0$,

$$\hat{y}_0 \pm t_{\alpha/2, n-p} \times \sqrt{MSE + se(\hat{y}_0)^2}$$

where $\sqrt{MSE + se(\hat{y}_0)^2}$ is the standard error of prediction, and $t_{\alpha/2, n-p}$ is the t-multiplier with the same degree of freedom as the mean square error $MSE$.

**Predict SLA violation**

Finally, the residual error between predicted and validated response time are calculated to check how well the fitted model predicts the response time in the future. Unless a rejection results the whole process going back to step one for model selection, the fitted model is valid and prediction will be accurate when the new observed response time lies within the 90% prediction interval and the residual errors are less
that 10%. At this time, the metamodel is ready to be used for quantitative analysis of the influence of different factors on service quality and for the predication of situations when SLA could be violated.

3.3 Experiments

3.3.1 Experiments Setup

To examine the effectiveness of the proposed approach, experiments have been conducted on a Web service that belongs to an e-commerce system. A typical e-commerce system is complicated and consists of a variety of modules, such as search engine, order processing, payment, discount tools, etc. The Web service on which experiments will be performed is one of the components of the e-commerce system and is capable of generating a list of top ten webpages linking to different e-commerce websites with the lowest prices for certain products. It can be used to help users compare one item’s price sold in this e-commerce website with that of other e-commerce websites and make a reasonable price policy. This Web service uses the ASP.NET Web API and consists of a product controller, a thread manager, a cache manager, product services, and product DAL. The experiments were implemented by employing .NET Framework 4.5, Visual Studio 2017 Professional and SQL Server 2012 on Windows 10 Professional with Inter(R) Core(TM) i5-4210U CPU and 8G RAM.

3.3.2 Design and Perform Experiments

Among the candidate factors that may influence the quality of this Web service in terms of response time, three are considered potentially significant, i.e., the number of sources, the number of products and the number of users to search, which are labeled A, B and C respectively in Table 3.1. As the five, five and three different levels of
3. SENSITIVITY AND METAMODEL-BASED ANALYSES FOR SERVICES DESIGN

Table 3.1: Data Collection

<table>
<thead>
<tr>
<th>B</th>
<th>C</th>
<th>50</th>
<th>100</th>
<th>150</th>
<th>200</th>
<th>250</th>
</tr>
</thead>
<tbody>
<tr>
<td>10,000</td>
<td>1</td>
<td>35.702</td>
<td>67.2039</td>
<td>87.605</td>
<td>119.807</td>
<td>142.508</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>35.621</td>
<td>68.9726</td>
<td>87.8588</td>
<td>119.689</td>
<td>145.796</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>35.8181</td>
<td>67.9856</td>
<td>85.1732</td>
<td>117.995</td>
<td>143.657</td>
</tr>
<tr>
<td>20,000</td>
<td>1</td>
<td>52.103</td>
<td>95.7055</td>
<td>135.408</td>
<td>189.911</td>
<td>242.014</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>54.2969</td>
<td>100.978</td>
<td>132.662</td>
<td>192.722</td>
<td>240.799</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>53.4769</td>
<td>96.8925</td>
<td>131.261</td>
<td>189.251</td>
<td>241.956</td>
</tr>
<tr>
<td>30,000</td>
<td>1</td>
<td>60.4034</td>
<td>114.507</td>
<td>172.21</td>
<td>232.113</td>
<td>297.317</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>59.154</td>
<td>112.976</td>
<td>174.699</td>
<td>235.977</td>
<td>295.266</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>58.9963</td>
<td>114.677</td>
<td>175.118</td>
<td>231.928</td>
<td>297.665</td>
</tr>
<tr>
<td>40,000</td>
<td>1</td>
<td>82.2047</td>
<td>138.208</td>
<td>220.113</td>
<td>294.717</td>
<td>353.92</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>83.6584</td>
<td>133.922</td>
<td>221.019</td>
<td>292.562</td>
<td>352.827</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>82.9885</td>
<td>139.655</td>
<td>217.923</td>
<td>293.379</td>
<td>356.284</td>
</tr>
<tr>
<td>50,000</td>
<td>1</td>
<td>85.6049</td>
<td>157.809</td>
<td>259.715</td>
<td>340.119</td>
<td>402.923</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>87.2549</td>
<td>160.143</td>
<td>255.222</td>
<td>340.876</td>
<td>401.116</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>84.2962</td>
<td>159.657</td>
<td>256.799</td>
<td>345.358</td>
<td>405.328</td>
</tr>
</tbody>
</table>

Factors A, B and C showing in the table generate 75 different combinations, a total of 150 experiments are conducted to collect two measurements for each combination. Provided in each cell of Table 3.1 is the average of observations in milliseconds.
Table 3.2: Variance Table

<table>
<thead>
<tr>
<th>Source</th>
<th>Sum of Square</th>
<th>DOF</th>
<th>Mean Square</th>
<th>F-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factor A</td>
<td>973897.464</td>
<td>4</td>
<td>243474.366</td>
<td>1592.790</td>
</tr>
<tr>
<td>Factor B</td>
<td>463802.886</td>
<td>4</td>
<td>115950.721</td>
<td>758.540</td>
</tr>
<tr>
<td>Factor C</td>
<td>480.904</td>
<td>2</td>
<td>240.452</td>
<td>1.573</td>
</tr>
<tr>
<td>Joint AB</td>
<td>111626.179</td>
<td>16</td>
<td>6976.636</td>
<td>45.640</td>
</tr>
<tr>
<td>Joint AC</td>
<td>2044.983</td>
<td>8</td>
<td>255.622</td>
<td>1.672</td>
</tr>
<tr>
<td>Joint BC</td>
<td>2222.770</td>
<td>8</td>
<td>277.846</td>
<td>1.817</td>
</tr>
<tr>
<td>Joint ABC</td>
<td>7155.858</td>
<td>32</td>
<td>223.620</td>
<td>1.462</td>
</tr>
<tr>
<td>Error</td>
<td>11464.517</td>
<td>75</td>
<td>152.860</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>1572695.561</td>
<td>149</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3.3.3 Perform Screen Design

The five-step process of sensitivity analysis produces a variance table in Table 3.2. For a confidence level of 95%, factor A has a significant impact on the response time as its $F$-value exceeds 2.50, the value of $F_{0.05,4,75}$, and so does factor B. However, factor C does not influence the response time significantly because its $F$-value is less than 3.13, the value of $F_{0.05,2,75}$. In addition, the response time of the Web service is largely affected by the joint effect of A and B due to the fact that the interaction $F$-value exceeds the value of $F_{0.05,16,75}$. The result of sensitivity analysis shows that the Web service is sensitive to the change of both factors A and B, and is insensitive to factor C. As a result, while factor C is removed from the factor list, factors A and B will be used as input factors to fit the surrogate model, with the experiments shown in Table 3.3 for the next process of metamodel-based analysis.
3. SENSITIVITY AND METAMODEL-BASED ANALYSES FOR SERVICES DESIGN

Table 3.3: Data Collection

<table>
<thead>
<tr>
<th>Factor</th>
<th>Factor A</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>50</td>
</tr>
<tr>
<td>10,000</td>
<td>35.702</td>
</tr>
<tr>
<td>20,000</td>
<td>52.103</td>
</tr>
<tr>
<td>30,000</td>
<td>60.4034</td>
</tr>
<tr>
<td>40,000</td>
<td>82.2047</td>
</tr>
<tr>
<td>50,000</td>
<td>85.6049</td>
</tr>
</tbody>
</table>

3.3.4 Specify One Appropriate Type of Metamodel

After performing sensitivity analysis to identify that both factor A and B have a significant influence on the response time, the next step is to perform experiments based on these two influential factors, to specify an appropriate type of metamodel and to fit its surrogate model. The experiment data in Table 3.3 will be used to fit the surrogate model. It can be found that the performance behavior in this case is low-order nonlinearity by an initial analysis of these experimental data. As a result, the polynomial regression model is selected as the metamodel type.

3.3.5 Specify a Series of Candidate Forms for the Metamodel and Construct These Surrogate Models

The six models listed in the second column of Table 3.4 become the candidate forms of polynomial regression model. For each of the six candidate forms, a fitting with the method of least squares [7] produces their $R^2$, $R^2_{adj}$, and $R^2_{pdt}$ values (Table 3.4). As quartic model yields the largest value of $R^2_{pdt}$, it is selected to predict the performance of the Web service, and the final approximating function of the polynomial regression model is formulated in (3.22) for response time ($rt$) in reference to
### Table 3.4: $R^2$, $R^2_{adj}$ and $R^2_{pdt}$ of fitted models

<table>
<thead>
<tr>
<th>No</th>
<th>Fitted Response Model</th>
<th>$R^2$</th>
<th>$R^2_{adj}$</th>
<th>$R^2_{pdt}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>linear</td>
<td>0.9253</td>
<td>0.9185</td>
<td>0.8904</td>
</tr>
<tr>
<td>2</td>
<td>2FI (Factor Interaction)</td>
<td>0.9935</td>
<td>0.9925</td>
<td>0.9888</td>
</tr>
<tr>
<td>3</td>
<td>quadratic</td>
<td>0.9949</td>
<td>0.9935</td>
<td>0.9893</td>
</tr>
<tr>
<td>4</td>
<td>cubic</td>
<td>0.9969</td>
<td>0.9950</td>
<td>0.9884</td>
</tr>
<tr>
<td>5*</td>
<td>quartic</td>
<td>0.9993</td>
<td>0.9984</td>
<td>0.9929</td>
</tr>
<tr>
<td>6</td>
<td>fifth</td>
<td>0.9999</td>
<td>0.9994</td>
<td>0.9920</td>
</tr>
</tbody>
</table>

the values of factors $A$ and $B$, where $a_A$ stands for the number of sources and $a_B$ for the number of products.

\[
rt = -106.35207 + 1.27249 \times a_A + 0.017516 \times a_B \\
-2.79588 \times 10^{-5} \times a_A \times a_B \\
-8.39782 \times 10^{-3} \times a_A^2 \\
-9.58326 \times 10^{-7} \times a_B^2 \\
+6.03267 \times 10^{-7} \times a_A^2 \times a_B \\
-7.03235 \times 10^{-10} \times a_A \times a_B^2 \\
+1.32674 \times 10^{-5} \times a_A^3 \\
+2.41739 \times 10^{-11} \times a_B^3 \\
-3.50836 \times 10^{-12} \times a_A^2 \times a_B^2 \\
-9.37387 \times 10^{-10} \times a_A^3 \times a_B \\
+1.64843 \times 10^{-14} \times a_A \times a_B^3 \\
+6.13368 \times 10^{-9} \times a_A^4 \\
-2.15762 \times 10^{-16} \times a_B^4 \quad (3.22)
\]
3. SENSITIVITY AND METAMODEL-BASED ANALYSES FOR SERVICES DESIGN

3.3.6 Check Model Adequacy

After fitting and identifying the polynomial regression model, several assumptions have to be checked to determine whether this fitted model is adequate.

![Normal Plot of Residuals](image)

**Figure 3.2: Normal Plot Of Residuals.**

To check whether residuals are normally distributed, a probability plot shown as Fig. 3.2 is used. By analyzing the normal plot of residuals, it is obvious that these points form a line. Consequently, a conclusion can be drawn that residuals come from a normal distribution.
3. SENSITIVITY AND METAMODEL-BASED ANALYSES FOR SERVICES DESIGN

To check whether the residual variance is constant, a plot with externally studentized residuals shown as Fig. 3.3 is used. It shows that the average of the residuals remains approximately 0 and the variation of the residuals appears to be roughly constant. As a result, residual variance is constant.

To check whether residuals are independent across the runs, a plot with residuals vs. Run shown as Fig. 3.4 is utilized. It shows that there is no strong linear or simple nonlinear trend in this plot. Therefore, residuals are independent across the runs.

In addition to residual normality, variance, autocorrelation, how accurate the fitted model is should be checked as well. The plot of predicted VS. actual shown as Fig. 3.5 is used, and it shows that this fitted model is accurate. Consequently, there is a strong correlation between the predicted response time of the fitted model and
3. SENSITIVITY AND METAMODEL-BASED ANALYSES FOR SERVICES DESIGN

3.3.7 Verify the Fitted Model

Once the model adequacy is validated, model confirmation should be performed to assess the accuracy of the prediction of the fitted model in the future. As a result, another round of ten different experiments produces new results as shown in Table 3.5, which indicates that the fitted model provides an accurate approximating function to the true function for predicting response time within 90% (90%−) prediction interval and the residual error relative to predicted values are less than 10% (90%+) in the future.

Figure 3.4: Residuals VS. Run.

corresponding actual observed results.
3.3.8 Analyse the Effects of Factors and their Interaction on Response Time

All the results of analyses indicate with consistent evidence that the selected model can be used for predicting the response time of the Web service under examination when the numbers of sources and products change due to structural or contextual variation of different SOA systems. As a result, this model can be used to analyze the effect of the number of sources, number of products and their interaction on response time and to predict SLA violation.

Graphically, a three-dimensional response surface plots as shown in Fig. 3.6 can be used to represent this fitted surrogate model and to help analyze and understand
### Table 3.5: Results of Verifying Experiments

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Actual</th>
<th>Predict</th>
<th>90%-%</th>
<th>90%+</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>75</td>
<td>20,000</td>
<td>70.904</td>
<td>72.410</td>
<td>62.883</td>
<td>81.938</td>
<td>1.506</td>
</tr>
<tr>
<td>75</td>
<td>50,000</td>
<td>123.807</td>
<td>119.225</td>
<td>109.375</td>
<td>129.075</td>
<td>-4.582</td>
</tr>
<tr>
<td>125</td>
<td>10,000</td>
<td>70.804</td>
<td>79.035</td>
<td>69.766</td>
<td>88.3054</td>
<td>8.231</td>
</tr>
<tr>
<td>125</td>
<td>30,000</td>
<td>142.308</td>
<td>140.003</td>
<td>131.254</td>
<td>148.752</td>
<td>-2.305</td>
</tr>
<tr>
<td>125</td>
<td>40,000</td>
<td>175.210</td>
<td>178.518</td>
<td>169.772</td>
<td>187.265</td>
<td>3.308</td>
</tr>
<tr>
<td>175</td>
<td>10,000</td>
<td>97.805</td>
<td>102.767</td>
<td>93.4969</td>
<td>112.036</td>
<td>4.962</td>
</tr>
<tr>
<td>175</td>
<td>20,000</td>
<td>162.009</td>
<td>163.903</td>
<td>155.156</td>
<td>172.65</td>
<td>1.894</td>
</tr>
<tr>
<td>175</td>
<td>40,000</td>
<td>248.414</td>
<td>256.237</td>
<td>247.49</td>
<td>264.984</td>
<td>7.823</td>
</tr>
<tr>
<td>225</td>
<td>30,000</td>
<td>269.315</td>
<td>267.621</td>
<td>259.688</td>
<td>275.554</td>
<td>-1.694</td>
</tr>
<tr>
<td>225</td>
<td>50,000</td>
<td>355.82</td>
<td>377.488</td>
<td>368.573</td>
<td>386.402</td>
<td>21.668</td>
</tr>
</tbody>
</table>

both the main and the interaction effects of these two factors. In terms of the number of sources, the change of the number of sources from 50 to 250 results in a significant increase in response time. This reflects the great effect of the number of sources on the response time. In terms of the number of products, the increase of the number of products from 10000 to 50000 results in a significant increase in response time as well. This shows the significant impact of the number of products on the response time. The interaction of these two factors have significant effects on the response time both in Fig. 3.6 and Fig. 3.7. Specifically, an increase in the number of sources results in the rise of response time and the rate of increase when the number of products increases from 10000 to 50000.

A further contour plot of the three-dimensional fitted surface model by the response time with respect to the two factors produces a map in Fig. 3.8. This figure shows that when both the number of sources and the number of products increase together, response time of the service has a tendency of going up from the blue region
Figure 3.6: Three-dimensional Response Surface Plot of the fitted model.

for 100 milliseconds or less to the red region for higher than 300 milliseconds. The significant increase of response time caused by the two factors is a clear indication of their impact on service performance.

Suppose that an SOA system requires in its SLA that the response time must be less than 50.0 milliseconds. An analysis with the fitted model predicts that the response time cannot meet the requirement when the number of sources increases from 50 to 250 or the number of products goes up from 10000 to 50000. To avoid the potential violation of SLA, the service provider may decide to either improve the service design or reconfigure the service. As the increase of response time is mainly caused by the time reading data from disk to memory when the number of sources and products starts to go up, the disk IO can be identified as the performance bottleneck. Accordingly, the service designer can make a decision to adapt caching strategy as an effective way to boost the performance among some candidate architectural tactics, rather than to choose provisioning more resources and maintaining multiple copies of
3. SENSITIVITY AND METAMODEL-BASED ANALYSES FOR SERVICES DESIGN

Figure 3.7: Interaction between the Number of Sources and the Number of Products with Respect to Response Time.

computation. This insight gained from performance analysis can help service designer avoid wasting time on the implementation of architectural tactics that may deliver small improvements with a lot of work.

3.3.9 Prediction Accuracy Comparison

The root mean squared error (RMSE) and mean absolute error (MAE) are used to measure and evaluate the accuracy of prediction. RMSE is defined as:
Figure 3.8: Contour plot between the number of sources and the number of products with respect to response time.

\[ RMSE = \sqrt{\frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{N}} \]  \hspace{1cm} (3.23)

MAE is defined as:

\[ MAE = \frac{\sum_{i=1}^{N} |y_i - \hat{y}_i|}{N} \]  \hspace{1cm} (3.24)

where \( y_i \) denotes the ith observed QoS value, \( \hat{y}_i \) is the corresponding predicted QoS value, and \( N \) is the number of predicted values.

A new series of experiments were conducted and 20 observations were collected to assess the prediction accuracy among 25%, 50% and 100% experimental regions.
3. SENSITIVITY AND METAMODEL-BASED ANALYSES FOR SERVICES DESIGN

Table 3.6: Comparison Based on Different Number of Factors

<table>
<thead>
<tr>
<th>Model</th>
<th>RMSE (25%)</th>
<th>MAE (25%)</th>
<th>RMSE (50%)</th>
<th>MAE (50%)</th>
<th>RMSE (100%)</th>
<th>MAE (100%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model 2</td>
<td>44.3053</td>
<td>35.2979</td>
<td>71.5077</td>
<td>56.4806</td>
<td>57.2106</td>
<td>43.5665</td>
</tr>
</tbody>
</table>

Then, the prediction accuracy were first compared among the following three models:

1. Model 1 is fitted by using three candidate factors without utilizing sensitivity analysis to remove the insignificant factors from the model.

2. Model 2 is fitted by selecting two factors randomly from three candidate factors and performing sensitivity analysis on these two factors.

3. Model 3 is fitted by using significant factors after performing sensitivity analysis on three candidate factors.

Results in Table 3.6 demonstrate that model 3 is able to consistently achieve higher accuracy among 25%, 50% and 100% experimental regions, and therefore outperforms the other models in term of prediction accuracy. The comparison between model 3 and model 1 shows that it is necessary to use sensitivity analysis to identify significant factors and ignore insignificant ones before doing metamodel-based performance analysis. It helps to solve the issue of overfitting the model and improve the prediction accuracy. Meanwhile, the comparison between model 3 and model 2 shows that it is necessary to improve the robustness of the sensitivity analysis method. In its original form [21], sensitivity analysis works with no more than two factors. As a result, two factors have to be randomly selected among three candidate factors, which happened in experiment to be the number of users and the number of sources. It causes the issue of underfitting the model because the number of products is a significant factor but not be used to fit the model. As a result, it is necessary to
Table 3.7: Comparison Based on Different Models

<table>
<thead>
<tr>
<th>Model</th>
<th>RMSE (25%)</th>
<th>MAE (25%)</th>
<th>RMSE (50%)</th>
<th>MAE (50%)</th>
<th>RMSE (100%)</th>
<th>MAE (100%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear Model</td>
<td>15.4784</td>
<td>12.7151</td>
<td>23.3908</td>
<td>16.7449</td>
<td>22.5625</td>
<td>17.2117</td>
</tr>
</tbody>
</table>

propose an approach of sensitivity analysis with high robustness as the one developed in this dissertation.

The value of $R^2_{pdt}$ determines how well a model will predict the response time for new observations. According to the largest $R^2_{pdt}$ value of the quartic model, it is selected as the most appropriate form of the metamodel to predict response time and expected to have a better prediction accuracy than the other four models. To verify this, another round of comparison of the prediction accuracy were made between a series of models, including linear model, factor interaction model, quadratic model, cubic model, and quartic model. All these models are fitted by using significant factors after performing sensitivity analysis on three candidate factors. The smallest RMSE and MAE values of the quartic model, shown in Table. 3.7, suggest that it outperforms the other models in terms of prediction accuracy. As a result, it is necessary to use the $R^2_{pdt}$ to select the most appropriate one from a series of candidate forms for the metamodel.
3.4 Summary

This chapter presents a framework that uses sensitivity analysis to identify influential input factors of services, produces a fitted model to approximate the relationship between factors and service quality, and uses this fitted model to analyze and predict QoS values. The process of QoS evaluation can be used by service consumers to select the most appropriate services when constructing an SOA system. Meanwhile, service providers can use the evaluation results to find out if and when SLA might be violated due to the fluctuation in different factors, to better understand the performance of a service with its fitted model, and to identify performance bottlenecks caused by influential factors. Future research will investigate automation of the proposed framework by using data mining and machine learning.
Chapter 4

A Strategy of QoS Optimization for Web Services

4.1 Introduction

The subject of QoS optimization can be either composite or individual services. For composite services, several approaches suggested to calculate aggregated QoS values of all possible service combinations and choose the one that maximizes the aggregated utility value while satisfying global constraints [89, 90, 91, 92]. Most of these methods are able to effectively and efficiently optimize QoS of the composite services. However, they fail to work for individual services. For individual services, resource provision has been utilized in [87] to analyze workloads, to categorize them on the basis of common patterns, and to plan for workloads before actual scheduling. The authors of [87] later enhanced their work with automated processing in [88]. Although these methods are able to optimize QoS by allocating more resources, no attention has been given to resource over-provision, which is a serious issue for Web services as it wastes resources and causes the increase of operational cost [38].

Presented in this chapter is a new approach of QoS optimization for quality
improvement of Web service. After briefly discussing the relationship between this chapter and the work presented in Chapter 3 in a unified framework for both prediction and prevention of SLA violation, Section 4.3 provides details of a novel strategy that helps service providers to achieve design optimum by searching globally to identify subregions with local optima and locally to locate the best optimum with constraints to minimize the operational cost. As an assessment of the proposed approach, Section 4.4 first uses multiple benchmark optimization problems to examine its accuracy, efficiency and robustness in comparison with existing methods, and then uses a Web service to check its efficiency in practice. Finally, this chapter concludes with discussions in Section 4.5.

4.2 Definition of the Optimization Problem and Related Conceptual Models

In this section, the optimization problem and conceptual models relevant to QoS of Web services are formally defined. These concepts will be used by the proposed optimization method in the following section.

4.2.1 Optimization Context Model

Optimization context model specifies the optimization objectives, the environmental context and the environment type. It can be modeled as a 3-tuple(Objectives, Factors, Environment), in which:

1. Objectives are QoS that need to be optimized, such as response time, security, etc.

2. Factors are those that may have a potential influence on the Objective. Each factor has its low and upper bound.
3. Environment represents the type of environment where Web services runs, such as cloud computing, traditional server, etc.

4.2.2 QoS Model

QoS model specifies the quality characteristic of objectives and can be modeled as a 3-tuple (Name, Type, Desirability), in which:

1. Name is the QoS name.
2. Typ is the data type of factors, such as numeric, string or boolean.
3. Desirability is the sense of desirability of QoS value that represents the direction in which the result is expected to go. It has three options: smaller is better, bigger is better or nominal is best.

4.2.3 Factor Model

Influential factors can be categorized as either uncontrollable or controllable ones [111]. Controllable factors refer to the ones that service providers can control or configure their level (value) during the execution of Web services, such as CPU, storage, cache, etc. Uncontrollable factors are those that can not be controlled by service providers, such as workload, network environment or the requirements of services consumer. For controllable factors, they can be further subdivided into two groups. One group includes factors that can cause an increase or decrease in the operational cost when changing their values. Another group contains factors that have no impact on the operational cost while their levels are modified. The objective of the proposed approach is to achieve design optimum without causing an unnecessary increase in the operational cost when reconfiguring Web services and optimizing its QoS. A factor can be modeled as a 4-tuple (Name, Type, Category, Cost), in which:
1. Name is the factor name.

2. Type is the data type of factor, such as numeric or string.

3. Category represents whether the factor is controllable or not.

4. Cost is an object that contains two properties: weight and influential type. Weight refers to the value of operational cost and its value ranges from 0 to 10. Influential type is a enumerable type that shows how this factor will influence operational cost. Its value can be either negative or positive.

4.2.4 SLA Model

An SLA specifies an agreement between service provider and service consumer and can be modeled as a 4-tuple (Name, QoS, RS, Penalty), in which:

1. Name is the SLA name.

2. QoS is the QoS model object that is used in the SLA.

3. RS is the average response time of Web services that the service provider promises to achieve.

4. Penalty is compensation that the service consumer will receive once the service provider fails to meet its promises.

4.2.5 Service Model

A service model specifies information relevant to Web Services and can be defined as a 3-tuple (Name, Factors, SLA), in which:

1. Name is the name of Web Services.
2. Factors are a collection of factors that have a significant influence on QoS.

3. SLA is the SLA model for this Web service.

4.2.6 Global Search Model

A global search model specifies information relevant to global search algorithm and can be defined as a 5-tuple (Factors, QoS, MARS, Knots, Results), in which:

1. Factors is a set of input variables of the global model, such as CPU, cache level, etc.
2. QoS is the output response (variable) of the global model, such as response time.
3. MARS is the fitted global model that reflects the relationship between one controllable factor and the response.
4. Knots is a collection of points that divide the region of the input variable into several subregions.
5. Results is a set of result objects that contain a candidate subregion in which a local optimum is located and a starting search point that is near the real local optimum. The results will be used by the local search algorithm.

4.2.7 Local Search Model

A local search model specifies information related to local search algorithm. For each result in the result list obtained from the global search, a corresponding local search model will be created. It can be modeled as a 6-tuple (Factor, QoS, Region, Model, Path, Result), in which:

1. Factor is the input variable of the local model, such as CPU, cache level, etc.
4. A STRATEGY OF QOS OPTIMIZATION FOR WEB SERVICES

2. QoS is the output response (variable) of the local model, such as response time.

3. Region is one of subregions that are obtained from the global search algorithm.

4. Model is the fitted local surrogate model that reflects the relationship between one controllable factor and the response. It is either a first or second order polynomial model.

5. Path refers to a searching path of the steepest ascent or descent computed by using the fitted first order model.

6. Result is the local optimum of one subregion.

4.2.8 Optimization Problem

Mathematically speaking, QoS optimization for Web services is the minimization of a function subject to constraints on the influential controllable factor and can be written as follows:

\[ \text{Minimize} : f(X) \]  
\[ \text{Subject to} : c_i(X) = 0, \ i \in E; c_i(X) > 0, \ i \in I \]  

where:

1. X refers to a set of influential controllable factors.

2. f(X) refers to the real relation between the controllable factor and QoS and its underlying formula is usually unknown. As a result, it is regard as a black box and the surrogate model will be used to approximate it.

3. E is a set of equality constraints.
4. A STRATEGY OF QOS OPTIMIZATION FOR WEB SERVICES

4. I is a set of inequality constraints.

4.3 SLA Violation Prevention with Optimization

This section discusses prediction and prevention of SLA violation with a unified framework, and provides details of QoS optimization with joint global search and local search algorithms.

4.3.1 A Unified Framework of QoS Optimization

Among a diverse range of approaches working on QoS optimization, sensitivity analysis, SLA violation prediction, and design optimization are three important yet normally separated techniques [67]. Presented in Fig. 4.1 is an innovative framework
4. A STRATEGY OF QOS OPTIMIZATION FOR WEB SERVICES

Table 4.1: M-Factor Variance Table

<table>
<thead>
<tr>
<th>source</th>
<th>mean square ($MS_{source}$)</th>
<th>f distribution ($F_{source}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A^{(1\cdots m)}$</td>
<td>$SS_i / \prod_{i=1}^{m} (a_i - 1)$</td>
<td>$MS_{A^{(1\cdots m)}} / MS_E$</td>
</tr>
<tr>
<td>error</td>
<td>$SS_E / [(n - 1) \prod_{i=1}^{m} a_i]$</td>
<td></td>
</tr>
<tr>
<td>total</td>
<td>$SS_T / [(n - 1) \prod_{i=1}^{m} a_i]$</td>
<td></td>
</tr>
</tbody>
</table>

that tackles the main issue of SLA violation by joining these three techniques into a unified process to analyze, predict, and prevent SLA violation. In the proposed framework, sensitivity analysis emphasizes on external observations of a service’s performance under the influence of different factors $A^{(v)}$, $1 \leq v \leq m$, that are beyond the control of the service provider. By developing a multi-factor model with a chosen experimental strategy, e.g. Taguchi’s orthogonal arrays [68], a variance table can be constructed from observations in the format of Table 4.1 for each of the individual factors and for all their joint effect in different combinations.

For all the $m$ uncontrollable factors, only those whose individual or joint $f$-distribution values exceed the threshold value of a cumulative $f$-distribution are considered to be influential factors of the service. After fitting an surrogate model, or metamodel, to external observations of the service with only the influential factors and verifying the model’s adequacy, accurate approximations of the service’s performance become accessible to predict the situations when SLA violation might happen and to prevent violation by means of QoS optimization. With details of sensitivity analysis and violation prediction being presented in Chapter 3, this chapter focuses on a new strategy of QoS optimization with controllable factors. This new strategy consists of a global search algorithm, which search globally to create a list of subregions with potential local optima, and a local search algorithm, which searches local optima to identify global optimum.
4.3.2 Global Search Algorithm

The aim of the proposed global search algorithm is to find one or more subregions in which potential local optima exist, and to obtain their corresponding starting search points that should be near the true local optimum in each subregion. These subregions will be used by the local search algorithm. Each subregion and its corresponding starting search point should have the following characteristics:

1. Each subregion must contain only one local optimum.

2. Each starting search point should be near the local optimum in order to increase the efficiency of the upcoming local search.

The global search algorithm will first fit a global surrogate model and then identify one or more subregions and their starting search points. The new algorithm chooses multivariate adaptive regression splines (MARS) for the following reasons:

1. It is capable of modelling both simple and complex performance model.

2. It can reflect the shape of real model of system accurately and help the global search algorithm to find subregions and their starting search points.

Specifically, the global search algorithm uses MARS to partition a given dataset into separate regions and fit each of them with a basis function [113]. With the basis function $H_i(X)$ taking the form of (4.3), $1 \leq i \leq n$, the overall model $\hat{f}(\hat{X})$ of MARS mainly consists of these $n$ basis functions as formulated in (4.4). In these two equations, $X$ is the factor vector; $K_i$ is a parameter introduced to limit the order of interactions; $s_{ki}$ assumes only two values that can be either 1 or $-1$ and indicates the left/right sense of the truncation; $[exp]_+$ produces zero if the value of exp is negative or its actual value otherwise; $t_{ki}$, known as the knot of a spline, has a value unique to $x_{v(k,i)}$; $B_0$ is an intercept parameter; and $B_i$ is the coefficient of $H_i(X)$. 
4. A STRATEGY OF QOS OPTIMIZATION FOR WEB SERVICES

\[ H_i(X) = \prod_{k=1}^{K_i} [s_{ki}(x_{v(k,i)} - t_{k,i})]_+ \]  
(4.3)

\[ f(\hat{X}) = B_0 + \sum_{i=1}^{n} B_i H_i(X) \]  
(4.4)

The approximation of a dataset with MARS starts with only \( B_0 \) being included in the model. After that, basis functions are created by iterating all factors and all possible knots, and those that give the maximum reduction in residual error will be added to the candidate list; This search-and-add activity continues recursively until reaching a model with pre-determined maximum complexity. Afterwards, a backward pruning procedure removes those basis functions that contribute the least to the approximation. Finally, its generalized cross validation error as introduced in [112] tells the adequacy of the fitted MARS model in terms of both its residual error and complexity.

In the proposed searching algorithm, the fitted global model is the relationship between one controllable factor and the performance. As a result, the characteristics of MARS model with one variable will be studied. According to [114], a MARS model in this case has the following characteristics:

1. An approximating spline function is obtained by dividing the range of \( x \) values into \( K + 1 \) range regions separated by \( K \) knots;

2. The approximation takes the form of a separate \( q \)th degree polynomial model in each region;

3. The flexibility of the regression spline approach can be enhanced by incorporating automatic strategy for knot selection as a part of the data fitting process.

Based on these characteristics, the global search algorithm, shown in Algorithm 1, is designed to find candidate subregions and their corresponding starting search points for the local search phase. With a list of knots created by fitting the dataset
Algorithm 1: The Global Search Algorithm

1. \text{MARSModel} \leftarrow \text{fit the MARS model with D};
2. \text{knots} \leftarrow \text{Get Knot list from MARSModel};
3. add the starting and ending point of D to knots and sort knots;
4. \text{R} \leftarrow \text{create an empty subregion list};
5. \text{IsCreateNewResult} \leftarrow \text{true};
6. \text{foreach knot of knots do}
   \hspace{1em} \text{if IsCreateNewResult is true then}
   \hspace{2em} \text{subRegion} \leftarrow \text{create a new subregion};
   \hspace{2em} \text{subRegion.StartingPoint} \leftarrow \text{knot};
   \hspace{2em} \text{IsCreateNewResult} \leftarrow \text{false};
   \hspace{1em} \text{else}
   \hspace{2em} \text{spline1, spline2} \leftarrow \text{get spline models of 2 subregions which are}
   \hspace{2em} \text{divided by this knot;}
   \hspace{2em} \text{if spline1 is monotonically increasing and spline2 is monotonically}
   \hspace{2em} \text{decreasing then}
   \hspace{3em} \text{subRegion.EndPoint} \leftarrow \text{knot};
   \hspace{3em} \text{IsCreateNewResult} \leftarrow \text{true};
   \hspace{3em} \text{add subRegion to R;}
   \hspace{1em} \text{end}
\hspace{1em} \text{end}
7. \text{foreach subRegion of R do}
   \hspace{1em} \text{subRegion.StartingSearchPoint} \leftarrow \text{Get } x_{\min}
\hspace{1em} \text{end}
8. \text{MinStartingSearchPoint} \leftarrow \text{Get the minimum starting Search Point from R;}
9. \text{foreach subRegion of R do}
   \hspace{1em} \text{diff} \leftarrow \text{abs(subRegion.StartingSearchPoint - MinStartingSearchPoint)};
   \hspace{1em} \text{if diff > Threshold, then}
   \hspace{2em} \text{Remove this subRegion from R;}
\hspace{1em} \text{end}
with a MARS model, the proposed global search algorithm completes its initialization by sorting the knot list after adding starting and ending points, creating an empty list of subregions, and setting flag IsCreateNewResult to true in line 3, 4, and 5 respectively. While the first loop (lines 6–18) adds subregions to \( R \), the second loop (lines 19–21) sets the starting points for all subregions in the list. Finally, the third loop (lines 23–27) removes some subregions in which the starting search points are far away from the global optimum to increase the searching efficiency for the following local search algorithm.

### 4.3.3 Local Search Algorithm

The local search algorithm will use the subregions and their starting search points identified by the global search algorithm, and iterate on each of these subregions. For each iteration, this local search algorithm will first start with its starting search point, fit one or more first order polynomial models, and use these models and the steepest descent or ascent method to get closer to new experimental regions that are closer to the real local optimum. When approaching to the real optimum on this subregion, one or more second order polynomial models will be used to optimize the response function. When there is no significant iteration-to-iteration improvement, the search on this subregion will come to an end and the point found will be regarded as a candidate point. After all subregions are searched, the best one will be used as the final solution among all candidate points.

In this approach, first or second order polynomial models are chosen as the local surrogate models and are given in Equations 4.5 and 4.6,

\[
\hat{y} = \beta_0 + \sum_{i=1}^{n} \beta_i x_i + \varepsilon \tag{4.5}
\]
\[
\hat{y} = \beta_0 + \sum_{i=1}^{n} \beta_i x_i + \sum_{i<j}^{n} \beta_{ij} x_i x_j + \sum_{i=1}^{n} \beta_{ii} x_i^2 + \varepsilon \quad (4.6)
\]

where \(\hat{y}\) is approximating function; \(x_i\) and \(x_j\) is corresponding input factors; \(\beta_0, \beta_i, \beta_{ii}\) and \(\beta_{ij}\) is unknown coefficients that need to be estimated; \(\varepsilon\) is Bias or lack of f.

When fitting the local surrogate models, design of experiments will be used in order to produce a good fit of the model to the data in a minimum number of experimental runs and control errors due to both variance and bias [7]. An experiment can be defined as a test or series of runs in which purposeful changes are made to the input variables of a process or system [8]. There are many types of experimental design, including a full factorial design (FFD), a central composite design (CCD), box-behnken design, the d-optimality, the space filling design and taguchi’s orthogonal arrays [7] [8] [115]. Because the proposed local search algorithm is developed by extending the response surface methodology, full factorial design will be used to fit the first order polynomial model and central composite design will be utilized to build the second order polynomial models [7].

After experiments based on full factorial design have been conducted and the first order polynomial model has been fitted, the steepest descent or ascent method will be used to find the searching path. The coordinates along the path of steepest descent or ascent depend on the signs and magnitudes of the regression coefficients in the fitted first order polynomial model [7]. Consider the following equation:

\[
L = \hat{y} + \lambda(\sum_{i=1}^{n} x_i^2 - r^2) = \beta_0 + \sum_{i=1}^{n} \beta_i x_i + \lambda(\sum_{i=1}^{n} x_i^2 - r^2) \quad (4.7)
\]

where \(\hat{y}\) is approximating function; \(x_i\) is corresponding input factors; \(\beta_0\) and \(\beta_i\) is coefficients; \(r\) is a fixed distance \(r\) from the center of the design; \(\lambda\) is a constant of proportionality.
4. A STRATEGY OF QOS OPTIMIZATION FOR WEB SERVICES

The derivative with respect to $x_i$ is

$$\frac{\partial L}{\partial x_i} = b_i - 2\lambda X_i = 0 \quad (4.8)$$

As a result, the coordinate of $x_i$ of the path of steepest descent or ascent is

$$x_i = \frac{b_i}{2\lambda} \quad (4.9)$$

When moving to the point that is closer enough to the optimum along the steepest descent or ascent path, experiments based on central composite design will be performed and the second order model will be fitted. After that, canonical and possible ridge analysis will be used to determine the location and the nature of the stationary point of the second order polynomial model [7]. The fitted second order polynomial model can be written in Equation 4.6.

$$\hat{y} = \beta_0 + X'b + X'\hat{B}X \quad (4.10)$$

where $\beta_0$, $b$, and $\hat{B}$ are the estimates of the intercept, linear, and second order coefficients, respectively.

The stationary point $x_s$ of the second-order polynomial is determined by

$$x_s = -\frac{1}{2}\hat{B}^{-1} \quad (4.11)$$

If all eigenvalues of $B$ are positive(negative), the quadratic surface has a minimum(maximum) at the stationary point $x_s$. If the eigenvalues have mixed signs, the stationary point $x_s$ is a saddle point and the search is in the direction of the negative gradient of the fitted quadratic, or may be a ridge direction which gives the best predicted value of the quadratic on a hypersphere of fixed radius [7].

Based on the above analysis, searching for optimum design will continue locally within each of the subregions that have been created by the global search algorithm. As illustrated in Algorithm 2, the algorithm repeats its search within each iteration
Algorithm 2: The Local Search Algorithm

1. $L \leftarrow$ create an empty list which is used to contain local optimum for each subregion;

2. **foreach** `subRegion` of $R$ **do**
3. fit a first-order model around the starting search point of this subregion;
4. **if** the controllable factor have negative influence on the operational cost and the performance on the starting search point meet the requirement of the SLA **then**
5. a path of steepest descent will be computed;
6. **else if** the controllable factor have positive influence on the operational cost and the performance on the starting search point meet the requirement of the SLA **then**
7. a path of steepest ascent will be computed;
8. **else**
9. a path of steepest descent will be computed;
10. **end**
11. perform a series of line searches along the computed path until a certain criteria is meet;
12. `finalPoint` $\leftarrow$ the last point along the searching path;
13. **if** the controllable factor influences on the operational cost and the performance on the final point meet the requirement of the SLA **then**
14. `localOptium` $\leftarrow$ `finalPoint`;
15. **else**
16. `secondOrderModel` $\leftarrow$ fit a second-order model around the final point;
17. `localOptium` $\leftarrow$ locate the stationary point of the fitted second-order polynomial model;
18. **end**
19. add `localOptium` to $L$;
20. **end**
21. get final optimum solution in $L$;

70
4. A STRATEGY OF QOS OPTIMIZATION FOR WEB SERVICES

(lines 2–18) by first fitting a first-order polynomial model and then following a path in the steepest descent or ascent direction to approach the regional optimum. When the flow reaches line 14, it is the case to find the local point by avoiding SLA violation while keeping the operational cost as low as possible at the same time. When the flow reaches line 16, it is the case to find the local optimum by using a second-order polynomial model for the optimization of response function. At the end of each iteration, a regional optimum is added to the candidate list for global optimum. When the algorithm finishes searching through all the subregions, the final optimum solution will be identified by getting the minimum value from the candidate list or the value that helps the service avoiding SLA violation while keeping the operational cost as low as possible at the same time.

4.4 Experiments

The effectiveness of one optimization algorithm against others cannot be simply measured by the problems that it solves if the set of problems are too specialized and without diverse properties [125]. As a result, this section uses experiments on both benchmark functions and a Web service to assess the efficiency, accuracy and robustness of the proposed optimization method.

4.4.1 Experiments with Benchmark Functions

Shown in Figure. 4.2 is the benchmark function of a well-known optimization problem [116], whose optimum is $f_{\min}(x^*) = -0.8690$ at $x^* = 0.5485$. For its unusual shape and multiple local optima as defined in (4.12), where $0.5 \leq x \leq 2.5$, this function is commonly used to test the accuracy and efficiency of optimization
4. A STRATEGY OF QOS OPTIMIZATION FOR WEB SERVICES

algorithms.

\[ f(x) = \frac{\sin(10\pi x)}{2x} + (x - 1)^4 \]  \hspace{1cm} (4.12)

Figure 4.2: The GRAMACY-LEE (2012) function.

Apply the Proposed Global and Local Search Algorithm

Full factorial design is used in the experiment to divide the design space of \( x \) into a series of uniformly spaces and the experimental data is collected to fit the global MARS model. As shown in Table. 4.2, a sequence of 100 values of the benchmark function \( f(x) \) is collected with \( x \) increasing by an increment of 0.02.
### Table 4.2: Result of Experiment for Global Search Algorithm

<table>
<thead>
<tr>
<th>No</th>
<th>x</th>
<th>f(x)</th>
<th>No</th>
<th>x</th>
<th>f(x)</th>
<th>No</th>
<th>x</th>
<th>f(x)</th>
<th>No</th>
<th>x</th>
<th>f(x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.50</td>
<td>0.0704</td>
<td>2</td>
<td>0.52</td>
<td>-0.5056</td>
<td>3</td>
<td>0.54</td>
<td>-0.8333</td>
<td>4</td>
<td>0.56</td>
<td>-0.8141</td>
</tr>
<tr>
<td>5</td>
<td>0.58</td>
<td>-0.4820</td>
<td>6</td>
<td>0.60</td>
<td>0.0176</td>
<td>7</td>
<td>0.62</td>
<td>0.4884</td>
<td>8</td>
<td>0.64</td>
<td>0.7573</td>
</tr>
<tr>
<td>9</td>
<td>0.66</td>
<td>0.7362</td>
<td>10</td>
<td>0.68</td>
<td>0.4490</td>
<td>11</td>
<td>0.70</td>
<td>0.0160</td>
<td>12</td>
<td>0.72</td>
<td>-0.3955</td>
</tr>
<tr>
<td>13</td>
<td>0.74</td>
<td>-0.6355</td>
<td>14</td>
<td>0.76</td>
<td>-0.6247</td>
<td>15</td>
<td>0.78</td>
<td>-0.3808</td>
<td>16</td>
<td>0.80</td>
<td>-0.0063</td>
</tr>
<tr>
<td>17</td>
<td>0.82</td>
<td>0.3529</td>
<td>18</td>
<td>0.84</td>
<td>0.5642</td>
<td>19</td>
<td>0.86</td>
<td>0.5557</td>
<td>20</td>
<td>0.88</td>
<td>0.3405</td>
</tr>
<tr>
<td>21</td>
<td>0.90</td>
<td>0.0080</td>
<td>22</td>
<td>0.92</td>
<td>-0.3129</td>
<td>23</td>
<td>0.94</td>
<td>-0.5033</td>
<td>24</td>
<td>0.96</td>
<td>-0.4977</td>
</tr>
<tr>
<td>25</td>
<td>0.98</td>
<td>-0.3062</td>
<td>26</td>
<td>1.00</td>
<td>-0.0079</td>
<td>27</td>
<td>1.02</td>
<td>0.2816</td>
<td>28</td>
<td>1.04</td>
<td>0.4547</td>
</tr>
<tr>
<td>29</td>
<td>1.06</td>
<td>0.4510</td>
<td>30</td>
<td>1.08</td>
<td>0.2785</td>
<td>31</td>
<td>1.10</td>
<td>0.0080</td>
<td>32</td>
<td>1.12</td>
<td>-0.2557</td>
</tr>
<tr>
<td>33</td>
<td>1.14</td>
<td>-0.4142</td>
<td>34</td>
<td>1.16</td>
<td>-0.4116</td>
<td>35</td>
<td>1.18</td>
<td>-0.2544</td>
<td>36</td>
<td>1.20</td>
<td>-0.0063</td>
</tr>
<tr>
<td>37</td>
<td>1.22</td>
<td>0.2367</td>
<td>38</td>
<td>1.24</td>
<td>0.3842</td>
<td>39</td>
<td>1.26</td>
<td>0.3843</td>
<td>40</td>
<td>1.28</td>
<td>0.2421</td>
</tr>
<tr>
<td>41</td>
<td>1.30</td>
<td>0.0160</td>
<td>42</td>
<td>1.32</td>
<td>-0.2056</td>
<td>43</td>
<td>1.34</td>
<td>-0.3389</td>
<td>44</td>
<td>1.36</td>
<td>-0.3352</td>
</tr>
<tr>
<td>45</td>
<td>1.38</td>
<td>-0.1985</td>
<td>46</td>
<td>1.40</td>
<td>0.0176</td>
<td>47</td>
<td>1.42</td>
<td>0.2315</td>
<td>48</td>
<td>1.44</td>
<td>0.3651</td>
</tr>
<tr>
<td>49</td>
<td>1.46</td>
<td>0.3728</td>
<td>50</td>
<td>1.48</td>
<td>0.2580</td>
<td>51</td>
<td>1.50</td>
<td>0.0704</td>
<td>52</td>
<td>1.52</td>
<td>-0.1137</td>
</tr>
<tr>
<td>53</td>
<td>1.54</td>
<td>-0.2212</td>
<td>54</td>
<td>1.56</td>
<td>-0.2088</td>
<td>55</td>
<td>1.58</td>
<td>-0.0792</td>
<td>56</td>
<td>1.60</td>
<td>0.1216</td>
</tr>
<tr>
<td>57</td>
<td>1.62</td>
<td>0.3226</td>
<td>58</td>
<td>1.64</td>
<td>0.4551</td>
<td>59</td>
<td>1.66</td>
<td>0.4785</td>
<td>60</td>
<td>1.68</td>
<td>0.3951</td>
</tr>
<tr>
<td>61</td>
<td>1.70</td>
<td>0.2480</td>
<td>62</td>
<td>1.72</td>
<td>0.1043</td>
<td>63</td>
<td>1.74</td>
<td>0.0291</td>
<td>64</td>
<td>1.76</td>
<td>0.0610</td>
</tr>
<tr>
<td>65</td>
<td>1.78</td>
<td>0.1986</td>
<td>66</td>
<td>1.80</td>
<td>0.4016</td>
<td>67</td>
<td>1.82</td>
<td>0.6070</td>
<td>68</td>
<td>1.84</td>
<td>0.7537</td>
</tr>
<tr>
<td>69</td>
<td>1.86</td>
<td>0.8050</td>
<td>70</td>
<td>1.88</td>
<td>0.7623</td>
<td>71</td>
<td>1.90</td>
<td>0.6640</td>
<td>72</td>
<td>1.92</td>
<td>0.5698</td>
</tr>
<tr>
<td>73</td>
<td>1.94</td>
<td>0.5382</td>
<td>74</td>
<td>1.96</td>
<td>0.6043</td>
<td>75</td>
<td>1.98</td>
<td>0.7675</td>
<td>76</td>
<td>2.00</td>
<td>0.9920</td>
</tr>
<tr>
<td>77</td>
<td>2.02</td>
<td>1.2214</td>
<td>78</td>
<td>2.04</td>
<td>1.4003</td>
<td>79</td>
<td>2.06</td>
<td>1.4956</td>
<td>80</td>
<td>2.08</td>
<td>1.5081</td>
</tr>
<tr>
<td>81</td>
<td>2.10</td>
<td>1.4720</td>
<td>82</td>
<td>2.12</td>
<td>1.4414</td>
<td>83</td>
<td>2.14</td>
<td>1.4693</td>
<td>84</td>
<td>2.16</td>
<td>1.5881</td>
</tr>
<tr>
<td>85</td>
<td>2.18</td>
<td>1.7976</td>
<td>86</td>
<td>2.20</td>
<td>2.0656</td>
<td>87</td>
<td>2.22</td>
<td>2.3411</td>
<td>88</td>
<td>2.24</td>
<td>2.5739</td>
</tr>
<tr>
<td>89</td>
<td>2.26</td>
<td>2.7332</td>
<td>90</td>
<td>2.28</td>
<td>2.8196</td>
<td>91</td>
<td>2.30</td>
<td>2.8640</td>
<td>92</td>
<td>2.32</td>
<td>2.9158</td>
</tr>
<tr>
<td>93</td>
<td>2.34</td>
<td>3.0235</td>
<td>94</td>
<td>2.36</td>
<td>3.2172</td>
<td>95</td>
<td>2.38</td>
<td>3.4969</td>
<td>96</td>
<td>2.40</td>
<td>3.8336</td>
</tr>
<tr>
<td>97</td>
<td>2.42</td>
<td>4.1807</td>
<td>98</td>
<td>2.44</td>
<td>4.4920</td>
<td>99</td>
<td>2.46</td>
<td>4.7393</td>
<td>100</td>
<td>2.48</td>
<td>4.9227</td>
</tr>
</tbody>
</table>
Table 4.3: The Fitted Mars Model for the Benchmark Function

<table>
<thead>
<tr>
<th>no.</th>
<th>basis functions</th>
<th>coefficient</th>
<th>no.</th>
<th>basis functions</th>
<th>coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Intercept</td>
<td>883.19</td>
<td>2</td>
<td>h(1.94-x)</td>
<td>-1025.41</td>
</tr>
<tr>
<td>3</td>
<td>h(x-2.1)*h(x-1.94)</td>
<td>-556.012</td>
<td>4</td>
<td>h(2.1-x)*h(x-1.94)</td>
<td>532.607</td>
</tr>
<tr>
<td>5</td>
<td>h(x-0.64)*h(1.94-0)</td>
<td>771.606</td>
<td>6</td>
<td>h(x-0.64)</td>
<td>-944.482</td>
</tr>
<tr>
<td>7</td>
<td>h(x-0.84)*h(x-0.64)</td>
<td>86.2546</td>
<td>8</td>
<td>h(0.84-x)*h(x-0.64)</td>
<td>-217.846</td>
</tr>
<tr>
<td>9</td>
<td>h(x-1.14)*h(x-0.94)</td>
<td>502.194</td>
<td>10</td>
<td>h(1.14-x)*h(x-0.94)</td>
<td>-380.269</td>
</tr>
<tr>
<td>11</td>
<td>h(1.24-x)*h(x-0.64)</td>
<td>-34.3123</td>
<td>12</td>
<td>h(x-1.54)*h(x-1.34)</td>
<td>93.1461</td>
</tr>
<tr>
<td>13</td>
<td>h(1.54-x)*h(x-1.34)</td>
<td>-42.5101</td>
<td>14</td>
<td>h(1.64-x)*h(x-0.64)</td>
<td>-21.9965</td>
</tr>
<tr>
<td>15</td>
<td>h(x-1.74)*h(1.94-x)</td>
<td>-225.029</td>
<td>16</td>
<td>h(1.74-x)*h(1.94-x)</td>
<td>297.525</td>
</tr>
<tr>
<td>17</td>
<td>h(x-1.44)*h(x-0.94)</td>
<td>-302.771</td>
<td>18</td>
<td>h(1.44-x)*h(x-0.94)</td>
<td>265.226</td>
</tr>
<tr>
<td>19</td>
<td>h(0.74-x)</td>
<td>259.016</td>
<td>20</td>
<td>h(x-2.34)*h(x-0.74)</td>
<td>151.82</td>
</tr>
<tr>
<td>21</td>
<td>h(2.34-x)*h(x-0.74)</td>
<td>-141.199</td>
<td>22</td>
<td>h(2.14-x)*h(x-0.94)</td>
<td>17.828</td>
</tr>
<tr>
<td>23</td>
<td>h(x-0.62)*h(0.94-x)</td>
<td>-114.808</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The MARS model is fitted by using this dataset with the pyearth module in Python as specified in (4.13) after setting maximum terms to 1000, the maximum degree to 2, and penalty to 0.01. This fitted global surrogate model contains 23 basis functions shown in Table. 4.3.

$$f(\hat{x}) = \sum_{i=1}^{23} C_i B_i$$  (4.13)

In (4.13), $B_i$ is a basis function and $C_i$ is the coefficient of the basis function in the $i_{th}$ row of Table 4.3. By applying the proposed global search algorithm, the whole dataset is divided into subregions as listed in Table 4.4 with their starting points to search for local optima. After setting the searching threshold to 0.2, only the first two subregions remain for further processing in the phase of local search while the
4. A STRATEGY OF QOS OPTIMIZATION FOR WEB SERVICES

Table 4.4: Subregions and Their Starting Search Points

<table>
<thead>
<tr>
<th>no.</th>
<th>subregions</th>
<th>starting points</th>
<th>( f(x_s) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1*</td>
<td>[0.5, 0.64]</td>
<td>0.54</td>
<td>-0.8333</td>
</tr>
<tr>
<td>2*</td>
<td>[0.64, 0.84]</td>
<td>0.74</td>
<td>-0.6355</td>
</tr>
<tr>
<td>3</td>
<td>[0.84, 1.14]</td>
<td>0.94</td>
<td>-0.5033</td>
</tr>
<tr>
<td>4</td>
<td>[1.14, 1.24]</td>
<td>1.14</td>
<td>-0.4142</td>
</tr>
<tr>
<td>5</td>
<td>[1.24, 1.44]</td>
<td>1.34</td>
<td>-0.3389</td>
</tr>
<tr>
<td>6</td>
<td>[1.44, 1.64]</td>
<td>1.54</td>
<td>-0.2212</td>
</tr>
<tr>
<td>7</td>
<td>[1.64, 2.10]</td>
<td>1.74</td>
<td>0.0291</td>
</tr>
<tr>
<td>8</td>
<td>[2.10, 2.50]</td>
<td>2.12</td>
<td>1.4414</td>
</tr>
</tbody>
</table>

others are abandoned.

Around the starting point of the first subregion, five \( \hat{f}(x) \) values are collected (Table. 4.5). Fitting data from no. 1 to 3 produces a fitted model in the form of

\[
\hat{f}(x) = 3.47043 - 7.91500 \times x.
\]

When searching along the path in the positive direction for two more increments of \( x \), no further improvement of performance is observed as the starting point is already near the local optimum. As a result, the surrounding values around the last searching point from no. 1 to no. 5 are used to fit the second-order model, which produces

\[
\hat{f}(x) = 129.16100 - 473.61437 \times x + 431.27143 \times x^2.
\]

A canonical and ridge analysis on this model delivers the local optimum in first subregion, i.e. \( f_{\min}(x^*) = -0.8680 \) at \( x^* = 0.55 \).

Similarly, the local optimum in the second subregion can be found with the proposed local search algorithm as \( f_{\min}(x^*) = -0.6627 \) at \( x^* = 0.75 \). As \( f_{\min}(0.55) \) is better than \( f_{\min}(0.75) \), the global optimum is achieved at 0.55 and the difference from the real \( x \) value is 0.01. The process of optimization terminates after performing 106 experiments.
Table 4.5: Result of Experiments on Local Search Algorithm

<table>
<thead>
<tr>
<th>no.</th>
<th>x</th>
<th>(f(x))</th>
<th>no.</th>
<th>x</th>
<th>(f(x))</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.53</td>
<td>-0.70971</td>
<td>2</td>
<td>0.54</td>
<td>-0.8333</td>
</tr>
<tr>
<td>3</td>
<td>0.55</td>
<td>-0.868</td>
<td>4</td>
<td>0.56</td>
<td>-0.8141</td>
</tr>
<tr>
<td>5</td>
<td>0.57</td>
<td>-0.6801</td>
<td>6</td>
<td>0.73</td>
<td>-0.544</td>
</tr>
<tr>
<td>7</td>
<td>0.74</td>
<td>-0.6355</td>
<td>8</td>
<td>0.75</td>
<td>-0.6627</td>
</tr>
<tr>
<td>9</td>
<td>0.76</td>
<td>-0.6247</td>
<td>10</td>
<td>0.77</td>
<td>-0.5271</td>
</tr>
</tbody>
</table>

Comparison of the Accuracy of Approximation and Prediction among Different Fitted Global Models

In this subsection, the error sum of squares (SSE) is used to measure the capability of how well the fitted global surrogate models approximate the true model and is defined as:

\[
SSE = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2
\]  \hspace{1cm} (4.14)

Meanwhile, the root mean squared error (RMSE) and mean absolute error (MAE) are utilized to measure and evaluate the accuracy of prediction. RMSE is defined as:

\[
RMSE = \sqrt{\frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{N}}
\]  \hspace{1cm} (4.15)

MAE is defined as:

\[
MAE = \frac{\sum_{i=1}^{N} |y_i - \hat{y}_i|}{N}
\]  \hspace{1cm} (4.16)

where \(y_i\) denotes the \(i\)th observed QoS value, \(\hat{y}_i\) is the corresponding predicted QoS value, and \(N\) is the number of predicted values.

The fitted MARS model will be compared with the following three commonly used modelling techniques in terms of SSE, RSME and MAE:
1. Polynomial regression (PR) [7]:

(a) Polynomial regression is the most commonly used modeling technique for its simplicity [117].

(b) The sklearn.linearmodel module in Python will be used to fit the polynomial regression model by setting the degree of the polynomial features to 10.

2. Support vector regression (SVR) [118]:

(a) Support vector regression is a modeling technique. It extends support vector machine to the domain of regression problems and gains in popularity in the machine learning community for its capability of producing prediction models with excellent generalization performance.

(b) The sklearn.svm module in Python will be used to fit the SVR model by setting the kernel function to the radial basis function, C (that is a penalty parameter of the error term) to 1e3 and gamma (that is the kernel coefficient for radial basis function to) to 0.1.

3. Regression tree (RT) [119]:

(a) Regression tree is a variant of decision tree that predicts values of continuous variables instead of labels of classification [120] and is frequently used to model non-linear relationships between input factors and output response.

(b) The sklearn.tree module in Python will be used to fit the RT model by setting the minimum samples’ split to 30, the minimum samples’ leaf to 10 and the random state to 0.

In comparison with these three modeling techniques regarding the accuracy of the global fitting, Table 4.6 shows that the MARS model has the smallest $SSE$ value,
Table 4.6: Comparison of Approximation and Prediction Accuracy of Fitted Global Models

<table>
<thead>
<tr>
<th>Model</th>
<th>SSE</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multivariate Adaptive Regression Splines</td>
<td>0.62255</td>
<td>0.06208</td>
<td>0.04611</td>
</tr>
<tr>
<td>Support Vector Regression</td>
<td>10.24481</td>
<td>0.29062</td>
<td>0.24912</td>
</tr>
<tr>
<td>Polynomial Regression</td>
<td>9.02551</td>
<td>0.27201</td>
<td>0.23638</td>
</tr>
<tr>
<td>Regression Tree</td>
<td>24.19414</td>
<td>0.45257</td>
<td>0.35261</td>
</tr>
</tbody>
</table>

which indicates that MARS model has the least error among all the fitted models and performs the best in terms of approximation accuracy. In addition, the smallest $RMSE$ and $MAE$ values of MARS suggest that it outperforms the other models in term of prediction accuracy.

By plotting the real model and all the four fitted models onto the graphs, Fig. 4.3 and Fig. 4.4 show that the MARS model fits the true model with higher accuracy than others and exhibits a similar shape as the true model across the global region.

![Comparison between Fitted Global Models](image)

Figure 4.3: The Comparison between All Fitted Global Models
4. A STRATEGY OF QOS OPTIMIZATION FOR WEB SERVICES

Figure 4.4: The Comparison Between the True Model and the Fitted Model
4. A STRATEGY OF QOS OPTIMIZATION FOR WEB SERVICES

Table 4.7: Optimization Results by Using Response Surface Methodology

<table>
<thead>
<tr>
<th>Starting Search Point</th>
<th>Number of Experiments</th>
<th>Optimum Design Point</th>
<th>Optimum Obtained</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.623</td>
<td>8</td>
<td>0.552</td>
<td>-0.8609</td>
<td>0.0080</td>
</tr>
<tr>
<td>0.791</td>
<td>5</td>
<td>0.751</td>
<td>-0.6618</td>
<td>0.2072</td>
</tr>
<tr>
<td>0.852</td>
<td>11</td>
<td>0.952</td>
<td>-0.5246</td>
<td>0.3444</td>
</tr>
<tr>
<td>1.227</td>
<td>9</td>
<td>1.147</td>
<td>-0.4326</td>
<td>0.43631</td>
</tr>
<tr>
<td>1.452</td>
<td>11</td>
<td>1.552</td>
<td>-0.2290</td>
<td>0.6399</td>
</tr>
</tbody>
</table>

Comparison with Several Well-known Optimization Methods

In this subsection, the proposed optimization method will be compared with response surface methodology (RSM) and the global metamodel-based optimization methodology. These comparisons will cover two aspects:

1. The optimization accuracy indicates the error between the identified and real global optimum. It tests how close the found global optimum comes to the real one;

2. The number of experiments tests the search efficiency of the optimization method. For benchmark optimization problem, performing one experiment refers to one benchmark function evaluation.

The first comparison was made with the response surface methodology [100], which is the most popular metamodel-based optimization method. The five rows in Table 4.7 are the optimization results starting with five different searching points by using the response surface methodology. Table 4.7 shows that the number of experiments needed by RSM is smaller than that of the proposed method. However, the optimization results are different from each other because RSM can only guarantee
Table 4.8: Optimization Results by Using Global Metamodel-based Optimization Methodology

<table>
<thead>
<tr>
<th>Model</th>
<th>Number of Experiments</th>
<th>Optimum Design Point</th>
<th>Optimum Obtained</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>MARS</td>
<td>106</td>
<td>0.550</td>
<td>-0.868</td>
<td>0.010</td>
</tr>
<tr>
<td>SVR</td>
<td>101</td>
<td>1.373</td>
<td>-0.251</td>
<td>0.617</td>
</tr>
<tr>
<td>PR</td>
<td>101</td>
<td>0.963</td>
<td>-0.477</td>
<td>0.391</td>
</tr>
<tr>
<td>RT</td>
<td>101</td>
<td>1.396</td>
<td>-0.023</td>
<td>0.845</td>
</tr>
</tbody>
</table>

to find a local optimum and has the issue of being trapped by the local optimum. As a result, the proposed method is more competitive than RSM in terms of finding the global optimum when there are multiple local optima in the performance model.

The second comparison was made with the global metamodel-based optimization methodology. For the global metamodel-based optimization methodology, three commonly used modeling techniques are used, including polynomial regression, support vector regression and regression tree. These three global models are fitted by using the result of 100 experiments in Table 4.2. Then, genetic algorithm (GA) is used to find the global optimum on these three surrogate models. While the first row in Table 4.8 shows the result of the proposed method, the other rows show results of PR, SVR and RT, respectively. The smallest error of the proposed method suggests that it outperforms the global metamodel-based optimization methodology that uses PR, SVR and RT, in terms of optimization accuracy.

The Robustness Evaluation of the Proposed Optimization Method

In previous subsections, the proposed method has been evaluated on the benchmark function with high degrees of non-linearity in terms of accuracy and efficiency.
A Strategy of QoS Optimization for Web Services

Table 4.9: Robustness Comparison of Approximation Models

<table>
<thead>
<tr>
<th>no.</th>
<th>point</th>
<th>opti.</th>
<th>error</th>
<th>no.</th>
<th>point</th>
<th>opti.</th>
<th>error</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td>2.885</td>
<td>-3.850</td>
<td>0.00030</td>
<td>37</td>
<td>7.958</td>
<td>-7.915</td>
<td>0.00163</td>
</tr>
<tr>
<td>20</td>
<td>2.888</td>
<td>-3.850</td>
<td>0.00043</td>
<td>34</td>
<td>7.960</td>
<td>-7.915</td>
<td>0.00124</td>
</tr>
<tr>
<td>20</td>
<td>2.846</td>
<td>-3.849</td>
<td>0.00048</td>
<td>34</td>
<td>7.991</td>
<td>-7.916</td>
<td>0.00066</td>
</tr>
<tr>
<td>20</td>
<td>3.127</td>
<td>-3.788</td>
<td>0.06200</td>
<td>34</td>
<td>6.658</td>
<td>-2.438</td>
<td>5.47811</td>
</tr>
</tbody>
</table>

\[
f_1(x) = -(16x^2 - 24x + 5)e^{-x}
\]

\[
f_2(x) = -x * \sin(x)
\]

\[
f_3(x) = \sin(x) + \sin(10x/3)
\]

\[
f_4(x) = -(1.4 - 3x)\sin(18x)
\]

Table 4.10: Details of the Four Additional Functions

<table>
<thead>
<tr>
<th></th>
<th>( f_1(x) )</th>
<th>( f_2(x) )</th>
<th>( f_3(x) )</th>
<th>( f_4(x) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>range of ( x )</td>
<td>[1.9, 3.9]</td>
<td>[2.7, 7.5]</td>
<td>[2.7, 7.5]</td>
<td>[0, 1.2]</td>
</tr>
<tr>
<td>( x^* )</td>
<td>2.868034</td>
<td>7.97875</td>
<td>5.145735</td>
<td>0.96609</td>
</tr>
<tr>
<td>( f_{\text{min}}(x^*) )</td>
<td>-3.85054</td>
<td>-7.916727</td>
<td>-1.899599</td>
<td>-1.48907</td>
</tr>
</tbody>
</table>

This subsection is going to evaluate its robustness and verify its ability to consistently achieve similar accuracies on different problems. This evaluation results in more experiments on four additional benchmark functions in different degrees of non-linearity. Going through the same procedure as for the first benchmark function in (4.12), more results are obtained and listed in Table 4.9. While the formula of each function is given in the table, results for the proposed method, polynomial regression, support vector regression, and regression tree are listed in the first, second, third, and
fourth row under each of the functions. The four benchmark functions have one, two, three, and four local optima respectively, with more details provided in Table 4.10. Underneath each function, results are listed in Table 4.9 in the order of the number of experiments, point of optimal design, the value of optimal design, and error between the real and fitted model.

Results in Table 4.9 demonstrate that the proposed method is able to consistently achieve high accuracy when applied to solve different problems, and therefore is robust. In comparison, metamodel-based optimization based upon polynomial regression and support vector regression may produce competitive results only when the benchmark functions have one, two, or three local optima. Once the number of local optima exceeds three, all the tested methods, except the proposed method, fail to achieve satisfying accuracy.

4.4.2 Experiments with A Web Service

Presented in this subsection is the second group of experiments that examines the proposed method about its use in the optimization of a Web service for users to identify the top ten webpages linking to different e-commerce Websites with the lowest prices for certain products.

Experiments Setup

For the implementation of Web services and collection of experimental data, the experiments use .NET Framework 4.5, Visual Studio 2017 Professional and SQL Server 2012 on Windows 10 Professional with Inter(R) Core(TM) i5-4210U CPU and 8G RAM. In addition, this group of experiments uses the python program language and the pyearth module for data analysis, the fitting of surrogate model, and performance optimization.
Optimization Planning and Analysis

Among different factors that influence the performance of this service, the number of sources and number of products to search are two factors beyond the control of the service provider as they vary from time to time and from application to application. In comparison, the cache level is a controllable factor as the service provider may use it to improve performance by loading product data into memory for fast processing.

The number of sources and the number of products have been identified in Chapter 3 as influential factors that cause SLA violation. For example, it is a violation of SLA if the required response time is less than 50 milliseconds because the Web service takes 77.8801 milliseconds when searching 40 sources for 10000 products. As
a result, the SLA is violated and the performance of this Web service needs to be optimized.

**Optimization Process**

To optimize the Web service using the proposed method, a series of experiments are first performed based upon full factorial designs. By keeping the two factors
constant at 10000 and 40 while changing the cache level from 0 to 100 in an increment of 5, a total of 21 observations of response time are collected (Table. 4.11) and Fig. 4.5 shows the plotting graph. Using the same setting of experiments on the benchmark functions, the MARS model uses seven basis functions to fit the dataset (Table. 4.12), which allows the identification of two subregions (Table. 4.13), with the second one being abandoned after setting the threshold of the global search to 5.

Even though the global optimum is the local optimum in the subregion [0, 65], a forward search along the path of the fitted MARS model (Fig. 4.5) from the starting point will miss the optimum as it has already surpassed the optimum point. In fact, the higher the cache level is, the more the memory will be occupied by cache data, causing more operational costs. To allow the Web service to use less memory by lowering the cache level, the search for the optimum in practice goes backward from 60 in a decrement of 5. Due to the required 50 milliseconds, the search stops at 25 as it uses the least cache level to yield a response time of 45.7550. This decision of QoS optimization not only improves service performance but also avoids causing the cache data to occupy too much unnecessary memory.

### 4.5 Summary

As part of an overall framework developed to predict and prevent SLA violation, this chapter proposes a novel strategy that uses combined global and local searching to help service providers achieve design optimum without over-provision of resources. Experimental results demonstrate that the proposed method outperforms
other commonly used methods in search efficiency while retaining competitive with those methods in optimization accuracy.
Chapter 5

Conclusions and Future Works

5.1 Conclusions

This Ph.D. dissertation introduces a framework to integrate sensitivity analysis, SLA violation prediction, and SLA violation prevention into a unified process for quality improvement of Web services. This framework identifies influential input factors of services, produces a fitted model to approximate the relationship between factors and service quality, and uses this fitted model to analyze and predict QoS values. The process of QoS evaluation can be used by service consumers to select the most appropriate services when constructing an SOA system. Meanwhile, service providers can use the evaluation results to find out if and when SLA could be violated due to the fluctuation in different factors, to better understand the performance of any service with its fitted model, and to identify performance bottlenecks caused by influential factors.

When SLA violation is predicted by analysis or detected at run time, the optimization strategy presented in this proposed framework can be used to prevent SLA violation with a combined search of global optimum through the identification of local optima in subregions created by a fitted MARS model. A decision of opti-
mization based upon the type of controllable factors further allows Web services to be optimized with minimized operational cost. In addition to technical details, this dissertation also includes results from experiments on multiple benchmark optimization problems to demonstrate the accuracy and robustness and on a Web service to validate the efficiency of the proposed method.

5.2 Future Works

In the proposed framework for SLA prediction, a manual process is employed, which is time consuming. In some cases, SLA violation prevention has to be performed in real-time to detect violations quickly. Therefore, an automated technique is required. In addition, only one controllable factor (cache level) is used for optimization, which will not lead to optimization for complex system. Therefore, the optimization method will be extended to be capable of preventing the SLA violation by using multi-factors optimization.

Future research will investigate the automation of the proposed approach using machine learning techniques that can easily adapt to any complex system and overcome the limitation of these approaches. Meanwhile, directions of future research will also include experiments on more Web services or SOA applications and development of a working system for use in practice. The plan is:

1. A real-world dataset (Google Cloud Compute Trace) will be used to perform experiments. This dataset will first be analyzed to understand its characteristics, features, and class distribution. This can help discover effective models to ascertain the best response time prediction with such features.

2. After finishing the data analysis, different regression models will be fitted to predict the violation. From the evaluation results, the best-fitted model will be selected for prediction. Vector autoregressive moving average model with
exogenous variables will be used to predict SLA violations

3. Once the SLA violation is detected through the selected model, SLA violation will be avoided by considering a multi-factor technique for optimizing a service.
Bibliography


[49] J. Grundy, G. Ding, and J. Hosking. "Deployed software component testing using
dynamic validation agents,". Journal of Systems and Software: Special Issue on
Automated Component-based Software Engineering, Pages 5-14, 2005.


composing web services,". In Proc. 2005 ACM/IEEE International Conference

[54] C. Lung and K. Kalachivelan. "An approach to quantitative software architec-
ture sensitivity analysis,". In Proc. 10th International Conference on Software

and prediction of component oriented distributed systems,". In Proc. ACM 3rd

and security of software systems,". In Proc. 5th International Workshop on

Discipline,". Prentice-Hall, 1996.

atin Responsive, Scalable Software,". Addison-Wesley, 2002.

[59] B. Spiznagel and D. Garlan. "Architecture-based performance analysis,". In
Proc. International Conference on Software Engineering and Knowledge Engi-

with sensitivity analysis,". In Proc. 2006 IEEE International Conference on Elec-
atro/information Technology, 2006.

[61] D. Kluver, M. Ekstrand, and J. Konstan. "Rating-Based Collaborative Filtering:

time collaborative filtering algorithm,". Information Retrieval, Pages 133-151,


[74] S Wang, Y Zhao, L Huang, J Xu, CH Hsu, "QoS prediction for service recommendations in mobile edge computing,". Journal of Parallel and Distributed Computing, Available online, 2017.


Vita Auctoris

Peng Cheng has received BS and MS degrees in the PLA Information Engineering University, China, and Tongji University, China. He has 10 years of research and development experience in both the university and industry. He is currently a candidate for the Doctoral degree in Computer Science at the University of Windsor, Ontario and hopes to graduate in Fall 2019.