Chapter 1
Introduction

Abstract This chapter provides an overview of QoS prediction in cloud and service computing, including backgrounds, related works, and organizations of this book.

1.1 Overview

Cloud computing [6, 22] is a new type of Internet-based computing, whereby shared resources, software, and information are provided to computers and other devices on demand [38]. With the exponential growth of cloud computing as a solution for providing flexible computing resources, more and more cloud applications emerge in recent years. The architecture of the Software-as-a-Service (SaaS) systems in the delivering of cloud computing typically involves multiple cloud components communicating with each other over application programming interfaces, usually Web services. [92]. Cloud computing has become a scalable service consumption and delivery platform.

Web services are software systems designed to support interoperable machine-to-machine interaction over a network. The technical foundations of cloud computing include service-oriented architecture (SOA), which is becoming a popular and major framework for building Web applications in the era of Web 2.0 [63], whereby Web services offered by different providers are discovered and integrated over the Internet. Typically, a service-oriented system consists of multiple Web services interacting with each other over the Internet in an arbitrary way. In this book, service refers to Web service in service computing and cloud component which is delivered as a service in cloud computing.

Figure 1.1 shows the system architecture in cloud computing. In a cloud environment, the cloud provider holds a large number of distributed services (e.g., databases, servers, Web services), which can be provided to designers for developing various cloud applications. Designers of cloud applications can choose from a broad pool of distributed services when composing cloud applications. These services are usually invoked remotely through communication links and are dynamically integrated into the applications. The cloud application designers are located in different geographic

Y. Zhang and M.R. Lyu, *QoS Prediction in Cloud and Service Computing*,
SpringerBriefs in Computer Science, DOI 10.1007/978-981-10-5278-1_1

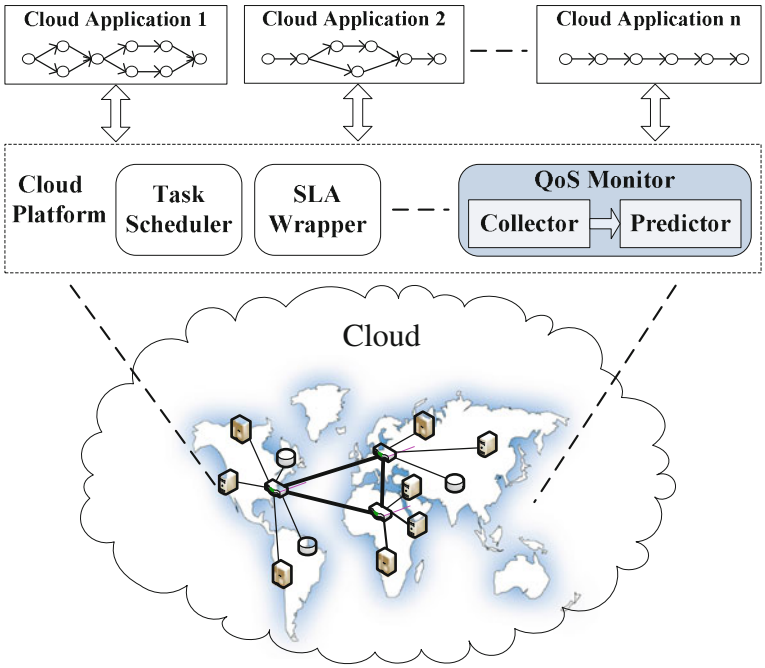

Fig. 1.1 System architecture. ©[2011] IEEE. Reprinted, with permission, from Ref. [104]

and network environments. Since the users invoke services via different communication links, the quality of services they observed are diverse.

Quality-of-Service (QoS) is usually employed to describe the non-functional characteristics of services. It becomes a major concern for application designers when making service selection [37]. Moreover, for the existing cloud applications, by replacing low-quality services with better ones, the overall quality of cloud applications can be improved.

In recent year, a number of research tasks have been focused on optimal service selection [10, 97] and recommendation [108] in distributed systems or service computing. Typically, evaluations on the service candidates are required to obtain their QoS values. In cloud environment, due to their various locations and communication links, different users will have different QoS experiences when invoking even the same service. Personalized QoS evaluation is required for each user at the user-side. However, a service user in general only invoked a limited number of services in the past and only received QoS performance information of these invoked services. In practice, therefore, conducting real-world evaluations on services to obtain their QoS information from the users' perspective is quite difficult, because (1) executing invocations for evaluation purposes becomes too expensive, since cloud providers who maintain and host services (e.g., Amazon EC2, Amazon S3) may charge for

invocations; (2) with the growing number of available services over the Internet, it is time-consuming and impractical to conduct QoS evaluations on all accessible services; (3) component users need to focus on building cloud applications rather than testing a large number of service candidates. Therefore, collecting historical usage records and conducting QoS prediction, which requires no additional invocation, is becoming an attractive approach. Based on the above analysis, in order to provide QoS information to application designers, we need to provide comprehensive investigation on QoS prediction approaches.

Employing the predicted QoS values, a QoS-aware Web service search engine can be enabled. Traditional Web service searching approaches only find the Web services to fulfill users' functionality requirements. However, Web services sharing similar functionalities may possess very different non-functionalities (e.g., response time, throughput, availability, usability, performance, integrity). Web services recommended by the traditional searching approach may not fulfill users' non-functional requirements. In order to find appropriate Web services which can fulfill both functional and non-functional requirements of users efficiently, QoS-aware searching approaches are needed.

Given the predicted QoS information, robust systems can be built based on redundant services by employing QoS-aware fault tolerance framework. Traditional fault tolerance framework [58] usually requires developing several different version of system services. However, due to the cost of development, the fault tolerance strategies are usually employed only for critical systems. In cloud computing, however, users can access multiple functional equivalent services via Internet at a very low cost. These services are usually developed and provided by different organizations, and can be dynamically composed to build a fault tolerance systems. Although some fault tolerance frameworks [52, 56, 107] have been proposed for traditional software systems, they cannot adopt to the highly dynamic cloud environment.

In order to provide accurate QoS prediction approaches, QoS-aware Web service searching mechanisms, and QoS-aware fault-tolerant frameworks for cloud systems, we proposed five approaches to attack these challenges in this book.

1.2 Backgrounds

1.2.1 QoS in Cloud and Service Computing

Cloud computing [6] has been in spotlight recently. Cloud computing has become a scalable service consumption and delivery platform [100]. The technical foundations of cloud computing include service-oriented architecture (SOA) [29]. SOA is becoming a popular and major framework for building Web applications in the era of Web 2.0 [63]. A number of investigations have been carried out focusing on different kinds of research issues such as Web service selection [28, 94, 97, 99], Web service composition [3, 4, 95], SOA failure prediction [9], SOA performance

prediction [105, 106], fault tolerance [52, 103], resiliency quantification [31], service searching [101], resource consistency [79], resource allocation [23], workload balance [87], dynamically resource management [46].

Quality-of-Service (QoS) has been widely employed as a quality measure for evaluating non-functional features of software systems and services [1, 97, 101]. A lot of research works have utilized QoS to describe the characteristics of services in cloud and service computing [42, 61, 64, 65, 72, 86]. Zeng et al. [98] use five QoS properties to compose Web service dynamically. Ardagna et al. [5] employ five QoS properties to conduct flexible service composition processes. Alrifai et al. [3] consider generic and domain-specific QoS for efficient service composition.

QoS performance of services can be measured either from the provider's perspective or from the user's observation. QoS values measured at the service provider side (e.g., price, availability) are usually identical for different users, such as QoS used in the service-level agreement (SLA) [57] (e.g., IBM [48] and HP [73]). While QoS values observed by different users may vary significantly due to the unpredictable communication links and heterogeneous contexts. In this book, we mainly focus on observing QoS data from users' perspective and making use of the QoS data for QoS prediction, service selection, service searching, and fault-tolerant framework building.

Based on the QoS performance of services, several approaches have been proposed for optimizing service selection [8, 10, 13, 27, 84, 97] in improving the whole quality of Web application, Web service composition [3, 5, 13, 14, 98], Web service recommendation [20, 86], reliability prediction [15, 21, 32, 35, 71], etc. Traditionally, reliability of a software system [59] is analyzed without considering the system performance, which is not accurate when applied to modern systems. Moreover, several QoS-aware approaches [24, 60, 72, 93, 97, 98] are proposed in cloud and service computing.

However, there is few real-world QoS data to verify these QoS-aware approaches. To collect the QoS data from the user-side, Zheng et al. [109] proposed a distributed evaluation framework and released the QoS datasets for further extensive research. Different from previous work [2, 89], they conduct large-scale real-world evaluations.

1.2.2 QoS Prediction in Cloud and Service Computing

The QoS-aware approaches usually assume that the QoS values are already known, while in reality a user cannot exhaustively invoke all the services. Although there existed some QoS evaluation approaches and publicly released QoS datasets, it is impossible to conduct personalized evaluation on all accessible services for all users. In this chapter, we focus on predicting missing QoS values by collaborative filtering approach to enable the QoS-aware approaches.

Collaborative filtering approaches are widely adopted in commercial recommender systems [12]. Generally, traditional recommendation approaches can be categorized into two classes: memory-based and model-based. Memory-based approaches, also known as neighborhood-based approaches, are one of the most popular prediction methods in collaborative filtering systems. Memory-based methods employ similarity computation with past usage experiences to find similar users and services for making the performance prediction. The typical example of memory-based collaborative filtering includes user-based approaches [11, 19, 39, 45, 81], item-based approaches [25, 43, 54, 78], and their fusion [34, 90]. Typically, memory-based approaches employ the PCC algorithm [70] for similarity computation.

Model-based approaches employ machine learning techniques to fit a predefined model based on the training datasets. Model-based approaches include several types: the clustering models [96], the latent factor models [74], the aspect models [40, 41, 82, 83]. Lee et al. [50] presented an algorithm for nonnegative matrix factorization that is able to learn the parts of facial images and semantic features of text. It is noted that there is only a small number of factors influencing the service performance in the user-service matrices, and that a user's factor vector is determined by how much each factor applies to that user. For a set of user-service matrices data, three-dimensional tensor factorization techniques are employed for item recommendation [69].

The memory-based approaches employ the information from similar users and services for predicting missing values. When the number of users or services is too small, similarity computation for finding similar users or services is not accurate. When the number of users or services is too large, calculating similarity values for each pair of users or services is time-consuming. In contrast, model-based approaches are very efficient for missing value prediction, since they assume that only a small number of factors influence the service performance.

There is few work of collaborative filtering prediction for QoS values in cloud and service computing, since there lack large-scale real-world QoS datasets for verifying the prediction accuracy. Some approaches [47, 85] employing a movie rating dataset (i.e., MovieLens [70]) for simulation. Shao et al. [80] only conduct a small-scale experiments, which involve 20 Web services for evaluating prediction accuracy.

The existing methods in the literature only consider two dimensions (i.e., user and Web service), while time factor is not included. The periodic features of service QoS values are ignored, which may improve the prediction accuracy significantly. Moreover, the high computational complexity makes it difficult to extend memory-based approaches to handle large amounts of time-aware performance data for timely prediction. There is a lack of fast algorithms to predict the QoS values at runtime to adapt the highly dynamic system environment in cloud and service computing.

In this book, we propose three approaches to address the QoS prediction problems in cloud and service computing, including memory-based prediction [104], time-aware prediction [102], and online prediction [105, 106] approaches. We also conduct large-scale real-world experiments to verify the prediction accuracy and release the QoS datasets for further studies of other researchers.

1.2.3 Web Service Searching

Web service discovery [68] is a fundamental research area in service computing. Several papers in the literature conduct investigations on discovering Web services through syntactic or semantic tag matching in a centralized UDDI repository [66, 88]. However, since UDDI repository is no longer a popular style for publishing Web services, these approaches are not practical now.

Text-based matching approaches have been proposed for querying Web service [33, 91]. These works employ term frequency analysis to perform keywords searching. However, most text descriptions are highly compact, and contain a lot of unrelated information to the Web service functionality. The performances of this approaches are not fine in practice. Plebani et al. [67] extract the information from WSDL files for Web service matching. By comparing with other works [26, 36, 44], it shows better performance in both recall and precision. However, it also does not consider non-functional qualities of Web services. Our searching approach, on the other hand, takes both functional and non-functional features into consideration.

Alrifai et al. [3], Liu et al. [55], and Tao et al. [97] focus on efficiently QoS-driven Web service selection. Their works are all based on the assumption: the Web service candidates are functional identical. Under this assumption, these approaches cannot be directly applied into Web service search engine. In this book, we proposed WSExpress [101], a QoS-aware searching approach which employs both QoS and functionality information, to search appropriate Web services for users.

1.2.4 Fault-Tolerant Cloud Applications

Software fault tolerance techniques (e.g., N-Version Programing [7], distributed recovery block [49]) are widely employed for building reliable systems [58]. Zhang et al. [101] propose a Web service search engine for recommending reliable Web service replicas. Salas et al. [75] propose an active strategy to tolerate faults in Web services. There are many fault tolerance strategies that have been proposed for Web services [17, 18, 30, 76]. Typically, the fault tolerance strategies can be divided into two major types: passive strategies and active strategies. Passive strategies include FT-CORBA [53], FT-SOAP [52]. Active strategies include WS-Replication [75], SWS [51], FTWeb [77].

However, these techniques cannot tolerate Byzantine faults like malicious behaviors. There are some works that focus on Byzantine fault tolerance for Web services as well as distributed systems. BFT-WS [107] is a Byzantine fault tolerance framework for Web services. Based on Castro and Liskov's practical BFT algorithm [16], BFT-WS considers client–server application model running in an asynchronous distributed environment with Byzantine faults. $3f + 1$ replications are employed in the server side to tolerate f Byzantine faults. Thema [62] is a Byzantine fault-tolerant (BFT) middleware for Web services. Thema supports three-tiered application model,

where the $3f + 1$ Web service replicas need to invoke an external Web service for accomplishing their executions. SWS [51] is a survivable Web service framework that supports continuous operation in the presence of general failures and security attacks. SWS applies replication schemes and N-Modular Redundancy concept. Each Web service is replicated into a service group to mask faults.

Different from above approaches, BFTCloud [103] proposed in this book aims to provide Byzantine fault tolerance for voluntary-resource cloud, in which Byzantine faults are very common. BFTCloud selects voluntary nodes based on both their reliability and performance characteristics to adapt to the highly dynamic voluntary-resource cloud environment.

1.3 Book Organization

As shown in Fig. 1.2, the rest of this book is organized as follows:

- Chapter 1
  This chapter briefly reviews some background knowledge and work related to the main methodology that will be explored in this book.
- Chapter 2
  In this chapter, we propose a novel neighborhood-based approach (CloudPred), which is enhanced by character modeling, for providing collaborative and personalized QoS prediction of cloud components. We first present the QoS prediction scenario by a toy example. Then, the QoS prediction problem in cloud computing is formally defined. After that, we present a latent feature learning algorithm to learn the user-specific and service-specific latent features. Based on the latent features, user and service similarity computation approaches are introduced. By identifying similar users and similar services to the active user-service pair, we formulate the CloudPred prediction Algorithm. We conduct extensive experiments to study the prediction accuracy of CloudPred and the impact of various parameters. The experimental results show that CloudPred achieves higher prediction accuracy than other competing methods.

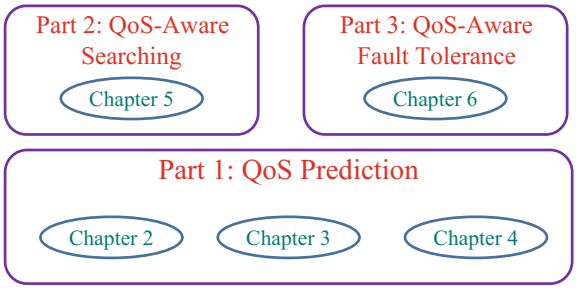

Fig. 1.2 Book structure

- Chapter 3
  In this chapter, we present a model-based time-aware collaborative filtering approach for personalized QoS prediction of Web services. First, we endow a new understanding of user-perspective QoS experiences, which is based on the following observations: (1) during different time intervals, a user has different QoS experiences on the same Web service; (2) in general, the differences are limited within a range. Based on these observations, we formulate the time-aware personalized QoS prediction problem as the tensor factorization problem, and propose an optimization formulation with average QoS constraint. Second, we propose to predict the missing QoS values by evaluating how the user, service, and time latent features are applied to each other. Furthermore, we provide a comprehensive complexity analysis of our approach, which indicates that our approach is efficient and can be applied to large-scale systems. Extensive experiments are conducted to evaluate the prediction accuracy and parameter impacts. The experimental results show the effectiveness and efficiency of our time-aware QoS prediction approach.
- Chapter 4
  In this chapter, we present an online Web service QoS prediction approach by performing time series analysis on user-specific and service-specific latent features. Our online prediction approach includes four phases. In Phase 1, service users monitor the performance of Web service and keep the QoS records in local site. In Phase 2, distributed service users submit local QoS records to the performance center in order to obtain a better QoS prediction service from the performance center. The performance center collects QoS records from different users and generates a set of global QoS matrices. In Phase 3, a set of time-stamped user latent feature matrices and service latent feature matrices are learned from the global QoS matrices. After that, time series analysis are conducted on the latent matrices to build a QoS model in the performance center. By evaluating how each factor applies to the active user and the corresponding service in the QoS model, personalized QoS prediction results can be returned to users on demand. In Phase 4, the system-level QoS performance of service-oriented architecture is predicted by analyzing the service compositional structure and utilizing the service QoS prediction results. The complexity analysis indicates that our approach is efficient and can be applied to large-scale online service-oriented systems. Finally, we conduct a number of experiments to study the performance of our approach and the impacts of algorithm parameters. We also study the effects of integrating service QoS information into the dynamic composition mechanism by a real-world service-oriented system case.
- Chapter 5
  In this chapter, we propose a QoS-aware Web service searching approach to explore the appropriate Web services to fulfill users' functional and non-functional requirements. We first describe the Web service searching scenarios and present the system architecture. Then, we present the QoS model to evaluate the non-functional utility of Web services. After that, functional similarity is introduced to evaluate the functional utility of Web services. Two QoS-aware Web service searching approaches are proposed: the score-based combination and the ranking-based combination. We

further extend the ranking-based approach to online searching scenario. Moreover, three common application scenarios are introduced. Finally, a number of experiments are conducted to study the functional and non-functional performance of our approach. The comprehensive results of experiments show that our approach provides better Web service searching results.

- Chapter 6
  This chapter presents a fault tolerance framework for building robust cloud applications at runtime. Our approach adopts dynamic QoS information to enable automatic system reconfiguration. We first introduce the architecture of our framework in voluntary-resource cloud. Then, we present the work procedures of our approach in detail, including 5 phases: primary selection, replicas selection, request execution, primary updating, and replica updating. After that, we conduct real-world experiments by deploying the prototype of our approach as a middleware in a voluntary-resource cloud environment, which consists of 257 distributed computers located in 26 countries. The experimental results show that our approach guarantees high reliability which enables good performance of cloud systems.
- Chapter 7
  The last chapter summarizes this book and provides some future directions that can be explored.

In order to make each of these chapters self-contained, some critical contents, e.g., model definitions or motivations having appeared in previous chapters, may be briefly reiterated in some chapters.

References

1. N. Ahmed, M. Linderman, J. Bryant, Towards mobile data streaming in service oriented architecture, in *Proceeding of SRDS'10*, pp. 323–327
2. E. Al-Masri, Q.H. Mahmoud, Investigating web services on the world wide web, in *Proceedings of the 17th International Conference on World Wide Web* (ACM, 2008), pp. 795–804
3. M. Alrifai, T. Risse, Combining global optimization with local selection for efficient QoS-aware service composition, in *Proceeding of International Conference on World Wide Web (WWW'09)* (2009), pp. 881–890
4. M. Alrifai, D. Skoutas, T. Risse, Selecting skyline services for QoS-based web service composition, in *Proceeding of WWW'10* (2010), pp. 11–20
5. D. Ardagna, B. Pernici, Adaptive service composition in flexible processes. IEEE Trans. Softw. Eng. **33**(6), 369–384 (2007)
6. M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica et al., A view of cloud computing. Commun. ACM **53**(4), 50–58 (2010)
7. A. Avizienis, The methodology of N-version programming, in *Software Fault Tolerance* (1995), pp. 23–46
8. W. Balke, M. Wagner, Towards personalized selection of web services, in *Proceeding of WWW'03* (2003)
9. C. Bird, N. Nagappan, H. Gall, B. Murphy, P. Devanbu, Putting it all together: using socio-technical networks to predict failures, in *Proceeding of ISSRE'09* (2009), pp. 109–119
10. P. Bonatti, P. Festa, On optimal service selection, in *Proceeding of WWW'05* (2005), pp. 530–538

11. J. Breese, D. Heckerman, C. Kadie, Empirical analysis of predictive algorithms for collaborative filtering, in *Proceedings of UAI'98* (1998), pp. 43–52
12. X. Cai, M. Bain, A. Krzywicki, W. Wobcke, Y. Kim, P. Compton, A. Mahidadia, Learning collaborative filtering and its application to people to people recommendation in social networks, in *Proceeding of ICDM'10* (2010), pp. 743–748
13. V. Cardellini, E. Casalicchio, V. Grassi, F. Lo Presti, Flow-based service selection for web service composition supporting multiple QoS classes, in *IEEE International Conference on Web Services* (2007), pp. 743–750
14. V. Cardellini, E. Casalicchio, V. Grassi, F. Lo Presti, R. Mirandola, QoS-driven runtime adaptation of service oriented architectures, in *Proceedings of the the 7th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering* (ACM, 2009), pp. 131–140
15. J. Cardoso, A. Sheth, J. Miller, J. Arnold, K. Kochut, Quality of service for workflows and web service processes. Web Semant. Sci. Serv. Agents. World Wide Web **1**(3), 281–308 (2004)
16. M. Castro, B. Liskov, Practical Byzantine fault tolerance. Oper. Syst. Rev. **33**, 173–186 (1998)
17. P. Chan, M. R. Lyu, M. Malek, Reliableweb services: methodology, experiment and modeling, in *IEEE International Conference on Web Services* (IEEE, 2007), pp. 679–686
18. P.P.W. Chan, M.R. Lyu, M. Malek, Making services fault tolerant, in Service Availability (Springer, Berlin, 2006), pp. 43–61
19. W. Chen, J. Chu, J. Luan, H. Bai, Y. Wang, E. Chang, Collaborative filtering for orkut communities: discovery of user latent behavior, in *Proceeding of WWW'09* (2009), pp. 681–690
20. X. Chen, X. Liu, Z. Huang, H. Sun, Regionknn: a scalable hybrid collaborative filtering algorithm for personalized web service recommendation, in *IEEE International Conference on Web Services* (IEEE, 2010), pp. 9–16
21. R.C. Cheung, A user-oriented software reliability model. IEEE Trans. Softw. Eng. **2**, 118–125 (1980)
22. M. Creeger, Cloud computing: an overview. ACM Queue **7**(5), 1–5 (2009)
23. A. Danak, S. Mannor, Resource allocation with supply adjustment in distributed computing systems, in *Proceeding of ICDCS'10* (2010), pp. 498–506
24. V. Deora, J. Shao, W. Gray, N. Fiddian, A quality of service management framework based on user expectations, in *Proceeding of ICSOC'03* (2003), pp. 104–114
25. M. Deshpande, G. Karypis, Item-based top-n recommendation algorithms. ACM Trans. Inf. Syst. (TOIS) **22**(1), 143–177 (2004)
26. X. Dong, A. Halevy, J. Madhavan, E. Nemes, J. Zhang, Similarity search for web services, in *Proceeding 30th International Conference on Very Large Data Bases (VLDB'04)* (2004), pp. 372–383
27. J. El Haddad, M. Manouvrier, G. Ramirez, M. Rukoz, QoS-driven selection of web services for transactional composition, in *Proceeding of ICWS'08* (IEEE, 2008), pp. 653–660
28. J. El Haddad, M. Manouvrier, M. Rukoz, Tqos: transactional and QoS-aware selection algorithm for automatic web service composition. IEEE Trans. Serv. Comput., 73–85 (2010)
29. T. Erl, *Service-oriented Architecture*, vol. 8 (Prentice Hall, New York, 2005)
30. H. Foster, S. Uchitel, J. Magee, J. Kramer, Model-based verification of web service compositions, in *IEEE International Conference on Automated Software Engineering* (IEEE, 2003), pp. 152–161
31. R. Ghosh, F. Longo, V. Naik, K. Trivedi, Quantifying resiliency of IaaS cloud, in *Proceeding of SRDS'10* (2010), pp. 343–347
32. S.S. Gokhale, K.S. Trivedi, Reliability prediction and sensitivity analysis based on software architecture, in *International Symposium on Software Reliability Engineering* (IEEE, 2002), pp. 64–75
33. K. Gomadam, A. Ranabahu, M. Nagarajan, A.P. Sheth, K. Verma, A faceted classification based approach to search and rank web apis, in *Proceeding of 6th International Conference on Web Services (ICWS'08)* (2008), pp. 177–184
34. S. Gong, A collaborative filtering recommendation algorithm based on user clustering and item clustering. J. Softw. **5**(7), 745–752 (2010)

35. V. Grassi, S. Patella, Reliability prediction for service-oriented computing environments. IEEE Internet Comput. **10**(3), 43–49 (2006)
36. Y. Hao, Y. Zhang, J. Cao, WSXplorer: searching for desired web services, in *Proceeding of 19th International Conference on Advanced Information System Engineering (CaiSE'07)* (2007), pp. 173–187
37. B. Hayes, Cloud computing. Commun. ACM **51**(7), 9–11 (2008)
38. T. Henzinger, A. Singh, V. Singh, T. Wies, D. Zufferey, FlexPRICE: flexible provisioning of resources in a cloud environment, in *Proceeding of CLOUD'10* (2010), pp. 83–90
39. J. Herlocker, J. Konstan, A. Borchers, J. Riedl, An algorithmic framework for performing collaborative filtering, in *Proceeding of SIGIR'99* (1999), pp. 230–237
40. T. Hofmann, Collaborative filtering via gaussian probabilistic latent semantic analysis, in *Proceeding of SIGIR'03* (2003), pp. 259–266
41. T. Hofmann, Latent semantic models for collaborative filtering. ACM Trans. Inf. Syst. (TOIS) **22**(1), 89–115 (2004)
42. M.C. Jaeger, G. Rojec-Goldmann, G. Muhl, Qos aggregation for web service composition using workflow patterns, in *IEEE International Enterprise Distributed Object Computing Conference* (IEEE, 2004), pp. 149–159
43. M. Jamali, M. Ester, Trustwalker: a random walk model for combining trust-based and item-based recommendation, in *Proceeding of KDD'09* (2009), pp. 397–406
44. Y. Jianjun, G. Shengmin, S. Hao, Z. Hui, X. Ke, A kernel based structure matching for web services search, in *Proceeding of 16th International Conference on World Wide Web (WWW'07)* (2007), pp. 1249–1250
45. R. Jin, J. Chai, L. Si, An automatic weighting scheme for collaborative filtering, in *Proceeding of SIGIR'04* (2004), pp. 337–344
46. G. Jung, M. Hiltunen, K. Joshi, R. Schlichting, C. Pu, Mistral: dynamically managing power, performance, and adaptation cost in cloud infrastructures, in *Proceeding of ICDCS'10* (2010), pp. 62–73
47. K. Karta, An investigation on personalized collaborative filtering for web service selection. *Honours Programme thesis, University of Western Australia, Brisbane, 2005*
48. A. Keller, H. Ludwig, The wsla framework: specifying and monitoring service level agreements for web services. J. Netw. Syst. Manag. **11**(1), 57–81 (2003)
49. K. Kim, H. Welch, Distributed execution of recovery blocks: an approach for uniform treatment of hardware and software faults in real-time applications. IEEE Trans. Comput. **38**(5), 626–636 (2002)
50. D. Lee, H. Seung, Learning the parts of objects by non-negative matrix factorization. Nature **401**(6755), 788–791 (1999)
51. W. Li, J. He, Q. Ma, I. Yen, F. Bastani, R. Paul, A framework to support survivable web services, in *Proceeding of IPDPS'05* (2005), p. 93b, 2005
52. D. Liang, C.-L. Fang, C. Chen, F. Lin, Fault tolerant web service, in *Tenth Asia-Pacific Software Engineering Conference* (IEEE, 2003), pp. 310–319
53. D. Liang, C.-L. Fang, S.-M. Yuan, C. Chen, G.E. Jan, A fault-tolerant object service on corba. J. Syst. Softw. **48**(3), 197–211 (1999)
54. G. Linden, B. Smith, J. York, Amazon. com recommendations: item-to-item collaborative filtering. IEEE Internet Comput. **7**(1), 76–80 (2003)
55. Y. Liu, A.H. Ngu, L.Z. Zeng, Qos computation and policing in dynamic web service selection, in *Proceeding of 13th International Conference on World Wide Web (WWW'04)* (2004), pp. 66–73
56. A. Luckow, B. Schnor, Service replication in grids: ensuring consistency in a dynamic, failure-prone environment, in *IEEE International Symposium on Parallel and Distributed Processing* (IEEE, 2008), pp. 1–7
57. H. Ludwig, A. Keller, A. Dan, R. King, R. Franck, A service level agreement language for dynamic electronic services. Electr. Commer. Res. **3**(1–2), 43–59 (2003)
58. M. Lyu, *Software Fault Tolerance*, (Wiley, 1995)
59. M. Lyu et al., Handbook of Software Reliability Engineering (1996)

60. E.M. Maximilien, M.P. Singh, Conceptual model of web service reputation. ACM Sigmod Rec. **31**(4), 36–41 (2002)
61. D.A. Menascé, Qos issues in web services. IEEE Internet Comput. **6**(6), 72–75 (2002)
62. M. Merideth, A. Iyengar, T. Mikalsen, S. Tai, I. Rouvellou, P. Narasimhan, Thema: Byzantine-fault-tolerant middleware for web-service applications, in *Proceeding of SRDS'05* (2005), pp. 131–140
63. T.O. Reilly, What is Web 2.0: design patterns and business models for the next generation of software. Commun. Strateg. **65**, 17 (2007)
64. J. O'Sullivan, D. Edmond, A. Ter Hofstede, What's in a service? Distrib. Parallel Databases **12**(2–3), 117–133 (2002)
65. M. Ouzzani, A. Bouguettaya, Efficient access to web services. IEEE Internet Comput. **8**(2), 34–44 (2004)
66. M. Paolucci, T. Kawamura, T. R. Payne, K. P. Sycara, Semantic matching of web services capabilities, in *Proceeding of 1st International Semantic Web Conference (ISWC'02)*, (2002), pp. 333–347
67. P. Plebani, B. Pernici, Urbe: web service retrieval based on similarity evaluation. IEEE Trans. Knowl. Data Eng. **21**(11), 1629–1642 (2009)
68. S. Ran, A model for web services discovery with QoS. ACM Sigecom Exch. **4**(1), 1–10 (2003)
69. S. Rendle, L. Schmidt-Thieme, Pairwise interaction tensor factorization for personalized tag recommendation, in *Proceeding of WSDM'10* (2010), pp. 81–90
70. P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, J. Riedl, GroupLens: an open architecture for collaborative filtering of netnews, in *Proceeding of CSCW'94*, (1994), pp. 175–186
71. R.H. Reussner, H.W. Schmidt, I.H. Poernomo, Reliability prediction for component-based software architectures. J. Syst. Softw. **66**(3), 241–252 (2003)
72. S. Rosario, A. Benveniste, S. Haar, C. Jard, Probabilistic QoS and soft contracts for transaction-based web services orchestrations. IEEE Trans. Serv. Comput. **1**(4), 187–200 (2008)
73. A. Sahai, A. Durante, V. Machiraju, Towards automated sla management for web services, *Hewlett-Packard Research Report HPL-2001-310 (R. 1)* (2002)
74. R. Salakhutdinov, A. Mnih, Probabilistic matrix factorization. Adv. Neural Inf. Process. Syst. (NIPS) **20**, 1257–1264 (2008)
75. J. Salas, F. Perez-Sorrosal, M. Patiño-Martínez, R. Jiménez-Peris, WS-replication: a framework for highly available web services, in *Proceeding of WWW'06* (2006), pp. 357–366
76. N. Salatge, J. Fabre, Fault tolerance connectors for unreliable web services, in *Proceeding of IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'07)* (2007), pp. 51–60
77. G. T. Santos, L. C. Lung, C. Montez, Ftweb: a fault tolerant infrastructure for web services, in *IEEE International EDOC Enterprise Computing Conference* (IEEE, 2005), pp. 95–105
78. B. Sarwar, G. Karypis, J. Konstan, J. Reidl, Item-based collaborative filtering recommendation algorithms, in *Proceeding of WWW'01* (2001), pp. 285–295
79. D. Serrano, M. Patiño-Martínez, R. Jiménez-Peris, B. Kemme, An autonomic approach for replication of internet-based services, in *Proceeding of SRDS'08* (2008), pp. 127–136
80. L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie, H. Mei, Personalized QoS prediction for web services via collaborative filtering, in *Proceeding of ICWS'07* (2007), pp. 439–446
81. Y. Shi, M. Larson, A. Hanjalic, Exploiting user similarity based on rated-item pools for improved user-based collaborative filtering, in *Proceeding of Recsys'09* (2009), pp. 125–132
82. L. Si, R. Jin, Flexible mixture model for collaborative filtering. ICML **3**, 704–711 (2003)
83. P. Singla, M. Richardson, Yes, there is a correlation:-from social networks to personal behavior on the web, in *Proceeding of WWW'08* (2008), pp. 655–664
84. A. Soydan Bilgin, M. P. Singh, A daml-based repository for QoS-aware semantic web service selection, in *IEEE International Conference on Web Services* (IEEE, 2004), pp. 368–375
85. R.M. Sreenath, M.P. Singh, Agent-based service selection. Web Semant. Sci. Serv. Agents. World Wide Web **1**(3), 261–279 (2004)
86. N. Thio, S. Karunasekera, Automatic measurement of a QoS metric for web service recommendation, in *Software Engineering Conference* (IEEE, 2005), pp. 202–211

87. K. Tsakalozos, M. Roussopoulos, V. Floros, A. Delis, Nefeli: hint-based execution of workloads in clouds, in *Proceeding of ICDCS'10* (2010), pp. 74–85
88. K. Verma, K. Sivashanmugam, A. Sheth, A. Patil, S. Oundhakar, J. Miller, Meteor-s wsdi: a scalable p2p infrastructure of registries for semantic publication and discovery of web services. Inf. Technol. Manag. **6**(1), 17–39 (2005)
89. M. Vieira, N. Antunes, H. Madeira, Using web security scanners to detect vulnerabilities in web services, in *IEEE/IFIP International Conference on Dependable Systems and Networks* (IEEE, 2009), pp. 566–571
90. J. Wang, A. De Vries, M. Reinders, Unifying user-based and item-based collaborative filtering approaches by similarity fusion, in *Proceeding of SIGIR'06* (2006), pp. 501–508
91. Y. Wang, E. Stroulia, Semantic structure matching for assessing web service similarity, in *Proceeding of 1st International Conference on Service Oriented Computing (ICSOC'03)* (2003), pp. 194–207
92. Wikipedia, http://en.wikipedia.org/wiki/cloud_computing
93. G. Wu, J. Wei, X. Qiao, L. Li, A bayesian network based qos assessment model for web services, in *IEEE International Conference on Services Computing* (IEEE, 2007), pp. 498–505
94. P. Xiong, Y. Fan, M. Zhou, QoS-aware web service configuration. IEEE Trans. Syst. Man Cybern. Part A Syst. Hum. **38**(4), 888–895 (2008)
95. P. Xiong, Y. Fan, M. Zhou, A petri net approach to analysis and composition of web services. IEEE Trans. Syst. Man Cybern. Part A Syst. Hum. **40**(2), 376–387 (2010)
96. G. Xue, C. Lin, Q. Yang, W. Xi, H. Zeng, Y. Yu, Z. Chen, Scalable collaborative filtering using cluster-based smoothing, in *Proceeding of SIGIR'05* (2005), pp. 114–121
97. T. Yu, Y. Zhang, K. Lin, Efficient algorithms for Web services selection with end-to-end QoS constraints. ACM Trans. Web (TWEB) **1**(1), 6 (2007)
98. L. Zeng, B. Benatallah, A. Ngu, M. Dumas, J. Kalagnanam, H. Chang, QoS-aware middleware for web services composition. IEEE Trans. Softw. Eng. (TSE) **30**(5), 311–327 (2004)
99. L. Zhang, S. Cheng, C. Chang, Q. Zhou, A pattern-recognition-based algorithm and case study for clustering and selecting business services. IEEE Trans. Syst. Man Cybern. Part A Syst. Hum. **42**(1), 102–114 (2012)
100. L.-J. Zhang, J. Zhang, H. Cai, *Services Computing* (Springer, Berlin, 2007)
101. Y. Zhang, Z. Zheng, M. Lyu, WSExpress: a QoS-aware search engine for Web services, in *Proceeding of ICWS'10* (2010), pp. 91–98
102. Y. Zhang, Z. Zheng, M. Lyu, Wspred: a time-aware personalized QoS prediction framework for web services, in *Proceeding of IEEE Symposium on Software Reliability Engineering (ISSRE'11)* (2011), pp. 210–219
103. Y. Zhang, Z. Zheng, M.R. Lyu, Bftcloud: a byzantine fault tolerance framework for voluntary-resource cloud computing, in *IEEE International Conference on Cloud Computing (CLOUD)* (IEEE, 2011), pp. 444–451
104. Y. Zhang, Z. Zheng, M.R. Lyu, Exploring latent features for memory-based qos prediction in cloud computing, in *IEEE Symposium on Reliable Distributed Systems (SRDS)* (IEEE, 2011), pp. 1–10
105. Y. Zhang, Z. Zheng, M.R. Lyu, Real-time performance prediction for cloud components, in *IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops (ISORCW)* (IEEE, 2012), pp. 106–111
106. Y. Zhang, Z. Zheng, M.R. Lyu, An online performance prediction framework for service-oriented systems. IEEE Trans. Syst. Man Cybern. Syst. (TSMC) **44**(9), 1169–1181 (2014)
107. W. Zhao, BFT-WS: a Byzantine fault tolerance framework for web services, in *Proceeding of EDOC'07* (2008), pp. 89–96
108. Z. Zheng, Y. Zhang, M. Lyu, CloudRank: a QoS-driven component ranking framework for cloud computing, in *Proceeding of SRDS'10* (2010), pp. 184–193
109. Z. Zheng, Y. Zhang, M. Lyu, Distributed QoS evaluation for real-world web services, in *Proceeding of ICWS'10* (2010), pp. 83–90