INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

FDV: FUNCTIONALLY DIVERSE SERVICES FOR ROBUST SERVICE COMPOSITIONS

S. Suganya[1], T. Priyaradhikadevi [2]

[1]*Department of Computer Science, Mailam Engineering College, Mailam, Tamil Nadu, India,*
ssuganyacollegemylam@gmail.com
[2] *Department of Computer Science, Mailam Engineering College, Mailam, Tamil Nadu, India*

Abstract

Service composition provides a means of customized and flexible integration of service functionalities. Quality-of-Service (QoS) optimization algorithms select services to adapt workflows to the non-functional requirements of the user. With increasing number of services in a workflow, previous approaches fail to achieve enough reliability. Moreover, exorbitant ad-hoc re-planning is required to assign with service failures. The major problem with such sequential application of planning and re-planning is that it ignores the potential costs during the initial planning and they consequently are concealed from the decision maker. Our plan to overcome this problem is to compute a QoS optimized selection of service cluster that includes a sufficient number of backup services for each service employed. These backup services should be sufficiently distributed to prevent a task failure in case of, e.g., a network failure. To support the decision maker in the selection task, our multi-objective approach considers the possible restore costs directly in the initial composition. Our graphical user interface envisions the resulting QoS of the workflow and the location of the services to enable the decision maker to select compositions in line with risk preferences. We prove the benefits of our approach in our detailed evaluation.

Keywords*:* Service-Oriented Computing, QoS-aware Service Composition, Multi-Objective Optimization.

1. Introduction

In Service-Oriented Architectures (SOA), services are arranged in workflows to create new functionality. Providers offer services that in many cases perform the same task but with varying Quality-of-Service (QoS) levels. QoS-aware service selection algorithms can be used to optimize the QoS of a given workflow during runtime. These algorithms select for each task one service, by taking, e.g., the preferences of user towards the price and reliability of the resulting workflow into consideration. Moreover, the user may also specify constraints on the QoS of the workflow. Since the search space is exponential, heuristic algorithms are employed to compute solutions in a feasible amount of time.
An examples scenarios for services compositions. A meaningful combination of tasks (white circles) is defined as a workflow, and there are choices on which service (gray circle) to use for each task.
Our main goal is to support the human decision maker in selecting a set of services that best fits his/her needs. We intend to compute QoS optimized compositions that can sustain a certain number of service failures. For that purpose, we need an extended QoS model that covers the probabilistic nature of these service executions.

The compositions are executed in open service environments, where certain service combinations might be invalid.

Most service composition approaches neglect service failures and aggregate the reliability with other QoS attributes (e.g., price), by using a simple additive weighting scheme. The algorithms might further sacrifice reliability in order to optimize other QoS attributes.

Causes of failures are often not transient, e.g., imprecise interface description or hardware failures. Therefore service substitution has been discussed actively (e.g., [1]), probably activated after failures of immediate retries. When an invocation failure occurs, the erroneous service can be substituted ad-hoc by either searching for a functional similar service [2] or by invoking replanning [3], [4]. If the constraints on the response time are tight or services with very specific functionality crash, a substitutions might not be possible and, as a result, the workflow execution fails. In our previous approach [5] we discussed how to collect backup services offline during the selection phase to avoid these issues.

However, these methods neglect the physical location of the backup services. If a hardware failure occurs on, e.g., a server host or a network router, all services on the same host or datacenter are affected. Depending on the criticality of the application, physical distribution of the backup services has to be enforced, though this also limits the possibilities of QoS optimization.

If the substitution is successful, it will have an impact on the resulting QoS of the workflow; the execution will be delayed and, depending on the reimbursement policy, the price will also increase to a certain extent. For that purpose, the QoS model needs to cover the probabilistic nature of the QoS of the workflows in order to facilitate computations to predict the expected outcome as well as the outcome in the best and worst cases.

2. Related Works

The QoS vector Q of the workflow is computed according to the types of QoS attributes and the control flow of the workflow. A detailed description of this aggregation can be found. Many related approaches simplify the QoS optimization problem by treating it as a single-objective optimization problem (SOO).

In addition, environmental conditions should be discussed carefully. In classical cases, each service is deployed on a specific host that the provider owns or is provided by a hosting service. In such cases, users should be careful about and responsible for redundancy at the host-level by themselves. As host-level locations are likely to be linked with IP addresses and thus can be detected with some confidence. On the other hand, in recent cloud computing, datacenter and area are observed units of locations. Redundancy at the host level location is often assured by the cloud providers. Host-level locations are rather not observable from users, controlled by the cloud providers very dynamically. In such cases, it is not necessary for users to care about redundancy at the host level, nor possible to obtain transient host-level locations precisely. We therefore expect users to select a set of valid levels examined in the specific setting, and then consider preference on each level.

To group services by locations, it is necessary to obtain the location information of services. We expect that the information at the datacenter or area levels are available in most cases. It is because there are increasing demands on the location information, to be included in service level agreements, because it matters concerns on not only redundancy but also legal compliance issues.

On the basis of these selections, the dialog. In the upper right section, the ranked solutions with all QoS values are listed. When the user activates the checkbox of one or more solutions, their QoS are visualized in the diagram in the upper left section. In this way, the user may balance different solutions.

The lower left section visualizes the services in the workflow of each selected solution. The number of services for each task is indicated by the colors red, yellow, and green. The color green indicates two or more backup services, yellow indicates only one backup service, and red means no backup service is provided. The combo box “Redundancy level” corresponds to the filter levels, with “Host level” being the most strict level and “Independent level” ignoring physical dependencies. When the user highlights tasks in the workflow view, the services of the selected solutions are visualized on a world map in the lower right section. In this way, the user can judge whether to accept weaker QoS values to increase the physical distribution of the services.

Each service has a price, response time, and reliability QoS attribute. The response time and reliability of each service are taken from the QWS dataset [6-15], the price is generated 50% random and 50% correlated to these two values; a low response time and a high reliability result in a high price and vice versa. We apply varying constraints on price and response time.

A solution is invalid if it contains an invalid link or a QoS constraint is violated. Each service is also associated with a service host. We use 10 independent service providers available depending on the execution time. In order to get a realistic estimate of the uptime of a service host we have monitored 100 real services by pinging them every 1000ms over 24 hours, leading to 8.64 million measurements. Each virtual host is associated with a random real world service host. If the host is unavailable, or the simulated service crashes, the service invocation fails. Since a service host might provide services from different tasks, multiple tasks might be affected. If all services from the execution order fail, the entire workflow execution fails.
We evaluate 100 test cases for each data point and limit the runtime of each algorithm to 5000 ms. The solutions of each algorithm are executed within our simulator 1,000 times and the minimal, maximal, and average values of price and response time are recorded. Moreover, the average reliability is measured by taking the ratio of successful number of runs to the total number of runs.
Since the problem of finding the optimal service selection [16] is NP-hard, heuristic algorithms [17-22] have been investigated as a means of computing near optimal solutions. To improve the performance of service selection, some approaches attempt to prune the search space. In [23], global QoS-constraints are split up into local constraints for each single service task. Subsequently, the configuration with the locally best services is selected. In [24], services not part of the skyline, which may also include backup services, are removed. Moreover, their definition of dominance [25-30] is not sufficient in our setting, since services that are not part of the skyline might be part of the optimal solution, e.g. especially when only few functionally valid solutions exist.

Genetic algorithms have been successfully used for multi-objective optimization; see, e.g. [31], [32]. These approaches consider several QoS attributes based on a classical QoS model as an objective function of the optimization problem. Thus, it is not possible to use the risk profile of the various feasible compositions. The papers employ a multi-objective stochastic program to take probabilistic QoS values into account. Although they consider the worst-case scenarios of QoS, they do not consider service failures.

3. Methods

The most common QoS model for service selection is presented. Several QoS attributes are characterized and methods to aggregate the QoS are provided. In [33], a probabilistic QoS model is provided to compute the expected case of the QoS values. The authors apply this model for predicting the average QoS of the workflow through many executions. The computed QoS of both models are only valid if there are no service crashes. A failure-aware model is presented in [34]. This model takes into account additional failure recovery times. The authors infer the QoS by collecting past execution logs. In our study, we incorporate the reliability of a service in the computation of the expected QoS a priori.

Regarding functionally diverse services, the matching quality of the service links is incorporated into the utility function in [35]. The authors balance the compliance with the QoS values, not evaluating whether the computed solution is executable. Our approach considers assurance of executability as a hard constraint. We have introduced a genetic algorithm that leverages the functional clustering to find functionally valid service compositions. This approach neglects the physical location and the reliability of the services. Moreover, it provides no failure recovery. It investigates the gap between the conceptual and data structural level of service composition. It will support more deep understanding of executability, but they do not take QoS-attributes into account.

The use of clustering prior to the selection phase is described. The authors use a k-nearest neighbor algorithm to arrange the services in QoS clusters, without caring functional compliance. The authors of the EASY project arrange services in a functionality graph in order to facilitate an efficient service discovery. They define service compliance in a different way from ours and do not consider QoS attributes. We presented an approach to leverage service clustering in service planning, without the probabilistic QoS model.

In this paper we present the following contributions:

1) Our *functional clustering* that determines backup services for each service. It considers locations of the backup services into account so that they are evaluated in terms of single points of failures.

2) Our *probabilistic QoS model* helps predicting the expected outcome when backup services are employed. It allows to estimate the range of possible QoS values even if service crashes occur. It also includes different levels of locations so that preferences on failure tolerance can be reflected.
3) Our improved *service selection algorithm* SHUURI2 that is based on the multi-objective optimization (MOO) algorithm NSGA-II. It leverages the clustering and the QoS model to compute a set of feasible solutions, verifying that the services are functionally compatible. It also considers service locations to efficiently find more reliable solutions.
4) Our *graphical user interface* that enables the user to compare solutions and to visualize the physical locations of candidate services and their backup services. The decision maker chooses the workflow selection that best fits his / her needs in terms of QoS and physical redundancy.

An *evaluation* of SHUURI2 by comparing it with its predecessor SHUURI and other related MOO algorithms by using our *workflow simulator*.

4. Conclusion

We discussed our approach that supports decision makers in finding robust, QoS optimized service compositions. Our approach takes into account functionally diverse services as a consequence of an open environment. Physical locations of the services are considered to minimize the impact of hardware or network failures. This result in a larger number of backup services grouped according to the location. The functional clustering on the services enables algorithms to detect backup services easily and to determine physical dependencies. Apart from that, we developed a new QoS model that helps to predict the resulting QoS of a workflow by considering service failures during the initial selection phase. For each service, a set of possible backup services and the corresponding QoS are computed beforehand.

We evaluated the performance of our selection algorithm SHUURI2 in various problem scenarios. It outperforms its predecessor SHUURI and related multi objective optimization algorithms in all relevant measures, especially with growing problem complexity. Finally, we presented a graphical interface that enables the decision maker to compare and choose a solution that best fits his/her needs. He/she provides his/her preferences, risk attitude, and constraints regarding the physical distribution of the services and receives an ordered list of approximate pareto-optimal selections. The interface visualizes the physical location of the services and helps to determine single points of failure.

As a next step, we plan to apply our approach in service planning that computes workflow templates. By combining both approaches we can provide a flexible and QoS optimized solution for composing workflows automatically. Moreover, we intend to extend our approach to consider inter-service-dependent QoS attributes. In this way, the time and performance of preceding services are considered. Apart from that, the physical locations of the services could be estimated by their latency, which could be modeled. Another direction is to consider other kinds of dependencies, such as services rely on the same component service.

REFERENCES

[1] E. Al-Masri and Q. H. Mahmoud. Discovering the best web service. In *Proceedings of the 16th international conference on World Wide Web*, (WWW), pages 1257–1258. ACM, 2007.

[2] M. Alrifai and T. Risse. Combining Global Optimization with Local Selection for Efficient QoS-aware Service Composition. In *Proceedings of the 18th International Conference on World Wide Web*, (WWW), New York, NY, USA, 2009. ACM.

[3] M. Alrifai, D. Skoutas, and T. Risse. Selecting Skyline Services for QoS-based Web Service Composition. In *Proceedings of the 19th International Conference on World Wide Web*, (WWW), pages 11–20, New York, NY, USA, 2010. ACM.

[4] B. Benatallah, M. Dumas, Q. Z. Sheng, and A. H. H. Ngu. Declarative composition and peer-to-peer provisioning of dynamic web services. In *Proceedings of the 18th International Conference on Data Engineering (ICDE)*, 2002.

[5] G. Canfora, M. Di Penta, R. Esposito, and M. L. Villani. An Approach for QoS-aware Service Composition based on Genetic Algorithms. In *Proceedings of the Conference on Genetic and Evolutionary Computation (GECCO)*, 2005.

[6] G. Canfora, M. Di Penta, R. Esposito, and M. L. Villani. QoSAware Replanning of Composite Web Services. In *Proceedings of the IEEE International Conference on Web Services*, (ICWS), pages 121–129, Washington, DC, USA, 2005. IEEE Computer Society.

[7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6:182–197, 2002.
[8] J. J. Durillo and A. J. Nebro. jMetal: A Java Framework for Multi- Objective Optimization. *Advances in Engineering Software*, 2011.
[9] D. Guinard, V. Trifa, P. Spiess, B. Dober, and S. Karnouskos. Discovery and on-demand provisioning of real-world web services. In *IEEE Internationl Conference on Web Services, (ICWS)*, 2009.
[10] F. Ishikawa, S. Katafuchi, F.Wagner, Y. Fukazawa, and S. Honiden. Bridging the Gap Between Semantic Web Service Composition and Common Implementation Architectures. *IEEE International Conference on Services Computing (SCC)*, 2011.
[11] M. C. Jaeger and H. Ladner. Improving the QoS of WS Compositions Based on Redundant Services. In *Proceedings of the International Conference on Next Generation Web Services Practices (NWESP)*. IEEE Computer Society, 2005.
[12] A. Klein, F. Ishikawa, and B. Bauer. A Probabilistic Approach to Service Selection with Conditional Contracts and Usage Patterns. In *Proceedings of the 7th International Joint Conference on Service-Oriented Computing (ICSOC)*, pages 253–268, 2009.
[13] A. Klein, F. Ishikawa, and S. Honiden. Efficient Heuristic Approach with Improved Time Complexity for QoS-aware Service Composition. In *IEEE International Conference on Web Services (ICWS)*, 2011.
[14] A. Klein, F. Ishikawa, and S. Honiden. Towards Network-aware Service Composition in the Cloud. In *Proceedings of the 21st international conference on World Wide Web*, (WWW), pages 959–968, New York, NY, USA, 2012. ACM.
[15] M. Klusch, B. Fries, and K. P. Sycara. Automated semantic web service discovery with OWLS-MX. In *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 915–922, 2006.
[16] S. Kukkonen and J. Lampinen. GDE3: The third Evolution Step of Generalized Differential Evolution. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 443 – 450, 2005.
[17] N. B. Lakhal, T. Kobayashi, and H. Yokota. A Failure-Aware Model for Estimating and Analyzing the Efficiency of Web Services Compositions. In *Proceedings of the Pacific Rim International Symposium on Dependable Computing (PRDC)*, 2005.
[18] M. Laukkanen and H. Helin. Composing workflows of semantic web services. In *Proceedings of the Workshop on Web-Services and Agent-based Engineering*, 2003.
[19] F. L´ecu´e, A. Delteil, and A. L´eger. Optimizing Causal Link Based Web Service Composition. In *Proceedings of the 18th European Conference on Artificial Intelligence (ECAI)*, pages 45–49, 2008.
[20] K.-J. Lin, J. Zhang, and Y. Zhai. An efficient approach for service process reconfiguration in SOA with End-to-End QoS constraints. In B. Hofreiter and H. Werthner, editors, *(CEC)*, pages 146–153. IEEE Computer Society, 2009.
[21] N. B. Mabrouk, S. Beauche, E. Kuznetsova, N. Georgantas, and V. Issarny. QoS-aware Service Composition in Dynamic Service Oriented Environments. In *Proceedings of the 10th ACM/IFIP/USENIX Intl. Conf. on Middleware*, 2009.
[22] S. B. Mokhtar, D. Preuveneers, N. Georgantas, V. Issarny, and Y. Berbers. EASY: Efficient semantic service discovery in pervasive computing environments with QoS and context support. *Journal of Systems and Software*, 81(5):785–808, 2008.
[23] J. O'Sullivan, D. Edmond, and A. H. M. ter Hofstede. What's in a Service? *Distributed and Parallel Databases*, 12(2/3):117–133, 2002.
[24] M. Reyes and C. Coello Coello. Improving PSO-based Multi-objective Optimization Using Crowding, Mutation and ǫdominance. In C. Coello, A. Hern´andez, and E. Zitler, editors, *Third International Conference on Evolutionary MultiCriterion Optimization (EMO)*, volume 3410 of *LNCS*, pages 509–519, 2005.
[25] A. Strunk. Qos-aware service composition: A survey. In *Proceedings of the 2010 Eighth IEEE European Conference on Web Services*, (ECOWS), pages 67–74, 2010.
[26] V. X. Tran and H. Tsuji. QoS Based Ranking for Web Services: Fuzzy Approaches. In *Proceedings of the 2008 4th International Conference on Next Generation Web Services Practices*, (NWESP), pages 77–82, Washington, DC, USA, 2008. IEEE Computer Society.
[27] H. Wada, P. Champrasert, J. Suzuki, and K. Oba. Multiobjective Optimization of SLA-Aware Service Composition. In *Proceedings of the 2008 IEEE Congress on Services - Part I*, (SERVICES), Washington, DC, USA, 2008. IEEE Computer Society.
[28] F. Wagner, F. Ishikawa, and S. Honiden. Applying QoS-aware Service Selection on Functionally Diverse Services. In *ICSOC Workshop on Non-Functional Properties and SLA Management in Service-Oriented Computing (NFPSLAM-SOC)*, Lecture Notes in Computer Science, 2011.

[29] F. Wagner, F. Ishikawa, and S. Honiden. QoS-Aware Automatic Service Composition by Applying Functional Clustering. *IEEE International Conference on Web Services (ICWS)*, 2011.
[30] F. Wagner, A. Klein, B. Kl¨opper, F. Ishikawa, and S. Honiden. Multi-objective service composition with time- and inputdependent qos. *IEEE International Conference on Web Services (ICWS)*, 2012.
[31] F. Wagner, B. Kl¨opper, F. Ishikawa, and S. Honiden. Towards Robust Service Compositions in the Context of Functionally Diverse Services. In A. Mille, F. L. Gandon, J. Misselis, M. Rabinovich, and S. Staab, editors, *(WWW)*, pages 969–978. ACM, 2012.
[32] J. Wang and Y. Hou. Optimal Web Service Selection based on Multi-Objective Genetic Algorithm. In *Proceedings of the 2008 International Symposium on Computational Intelligence and Design (ISCID)*, Washington DC, USA, 2008. IEEE Computer Society.
[33] W. Wiesemann, R. Hochreiter, and D. Kuhn. A Stochastic Programming Approach for QoS-Aware Service Composition. In *Proceedings of the 2008 Eighth IEEE International Symposium on Cluster Computing and the Grid (CCGRID)*, pages 226–233, 2008.
[34] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Z. Sheng. Quality Driven Web Services Composition. In *Proceedings of the 12th International Conference on World Wide Web (WWW)*, pages 411–421. ACM, 2003.
[35] E. Zitzler and L. Thiele. Multiobjective Optimization Using Evolutionary Algorithms - A Comparative Case Study. In *Proceedings of the Conference on Parallel Problem Solving from Nature (PPSN)*, London, UK, 1998. Springer-Verlag.

A Brief Author Biography

S. Suganya is interested on Web Services, Software Engineering, Network Security and Mobile Computing. She published one paper in national level conference.

T. Priyaradhikadevi, is interested on Web Services, Software Engineering, Network Security and Mobile Computing. Department of computer science, Mailam Engineering College, Mailam.