Robust Multi-criteria Service Composition in Information Systems

Service compositions are used to implement business processes in different application domains. A quality of service (QoS)-aware selection of the service to be composed involves multiple, usually conflicting and possibly uncertain QoS attributes. This article presents a heuristic multi-criteria service selection approach that is designed to determine a Pareto frontier of alternative service selections in a reasonable amount of time. Taking into account the uncertainty of response times, the obtained service selections are robust with respect to the constrained execution time. The proposed solution approach is based on the Non-dominated Sorting Genetic Algorithm (NSGA)-II that is extended by heuristics that exploit problem specific characteristics of the QoS-aware service selection. The applicability of the solution approach is demonstrated by a simulation study.

DOI 10.1007/s12599-014-0325-5

The Authors

Dipl.-Inf. René Ramacher
Prof. Dr. Lars Mönch (✉)
Chair of Enterprise-wide Software Systems
Department of Mathematics and Computer Science
University of Hagen
Universitätsstraße 1
58097 Hagen
Germany
Lars.Moench@fernuni-hagen.de

Received: 2013-07-13
Accepted: 2014-02-05
Accepted after two revisions by the editors of the special focus.
Published online: 2014-04-16

This article is also available in German in print and via http://www.wirtschaftsinformatik.de: Ramacher R, Mönch L (2014) Robuste multikriterielle Dienstkomposition in Informationssystemen. WIRTSCHAFTSINFORMATIK. doi: 10.1007/s11576-014-0416-4.

1 Introduction

Service-oriented architectures (SOA) are emerging as the architecture paradigm used to implement business information systems (Papazoglou et al. 2008; Aier et al. 2011). Business processes are implemented based on software services that are accessible by using standard interfaces in an SOA. The services are hosted inside an enterprise or purchased from external service providers. The selection of the services to be composed is driven by functional requirements and the QoS. Facing a growing market of services and alternative service implementations, the challenge of a QoS-aware service selection is to identify the services to be composed in a way that predefined QoS requirements are met and QoS attributes are optimized (Bichler and Lin 2006). We expect future applications of QoS-aware service selection algorithms for decision support systems in domains such as value chains, i.e., the orchestration of a network of production and logistics activities, supply chain management, or multi-segment intermodal transportation within the physical internet.

The QoS-aware service selection is a multi-criteria optimization problem that is NP-hard as shown by Yu and Lin (2004). Taking an economic point of view, the objective of a QoS-aware service selection is to minimize the cost of the service composition. According to Eder et al. (1999), customer satisfaction is achieved by a high availability of the business process and the compliance to temporal conditions during its execution. However, the execution time of the service composition is determined by the response times of the services which are arguably uncertain. The service reconfiguration approach proposed by Ramacher and Mönch (2013) is used to avoid a violation of a execution time limit in an uncertain execution environment. A frequent service replacement caused by a service reconfiguration is undesired in an environment where the service implementations and contracts with service providers cannot be adjusted on demand. A further drawback of existing service selection approaches is that the optimization of multiple QoS attributes is neglected. Typically, QoS attributes are combined to make up an integrated objective using a simple additive weighting (SAW) approach. However, the integration requires that a tradeoff between the QoS attributes is determined prior to the optimization and hence without information about possible alternatives.

This article presents a multi-criteria QoS-aware service selection model that considers the uncertainty of response times. According to Bertsimas and Sim (2004), a solution, i.e. a service selection, is called robust if it remains feasible with a certain probabilistic guarantee when the problem parameters change within a certain range. A robust service selection is obtained that ensures a reliable execution with respect to a constrained execution time and uncertain response times. Following Scholl (2001), this type of robustness is called robustness with respect to feasibility. Robust approaches have so far predominantly been neglected in the related literature. Adopting the principles of multi-criteria optimization, the

solution approach aims to generate a set of alternative solutions with respect to cost minimization and availability maximization objectives. The proposed solution approach combines the NSGA-II metaheuristic with heuristics that exploit the particular structure of the service selection problem. In the present article the heuristics proposed by Ramacher and Mönch (2012) are extended to deal with complex process models and uncertain response times. Computational experiments demonstrate that high-quality service compositions are determined in a few minutes of computing time while the computational burden caused by an exact solution approach increases to several hours just for medium-size service compositions.

The remainder of the article is organized as follows. The service selection problem is formulated in Sect. 2.1. Possible application domains of the QoS-aware service selection are presented in Sect. 2.2. Section 2.3 surveys related literature. Section 3.1 describes the metaheuristic-based solution approach for the examined multi-criteria service selection. Section 3.2 elaborates on problem-specific heuristics and their integration into the metaheuristic solution approach, while an exact approach is presented in Sect. 3.3. The design of the computational experiments is described, and the results are discussed in Sect. 4. Finally, Sect. 5 concludes the paper and provides some future research directions.

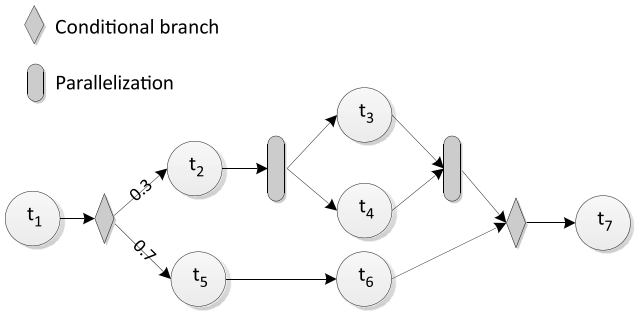

Fig. 1 Process model of a service composition

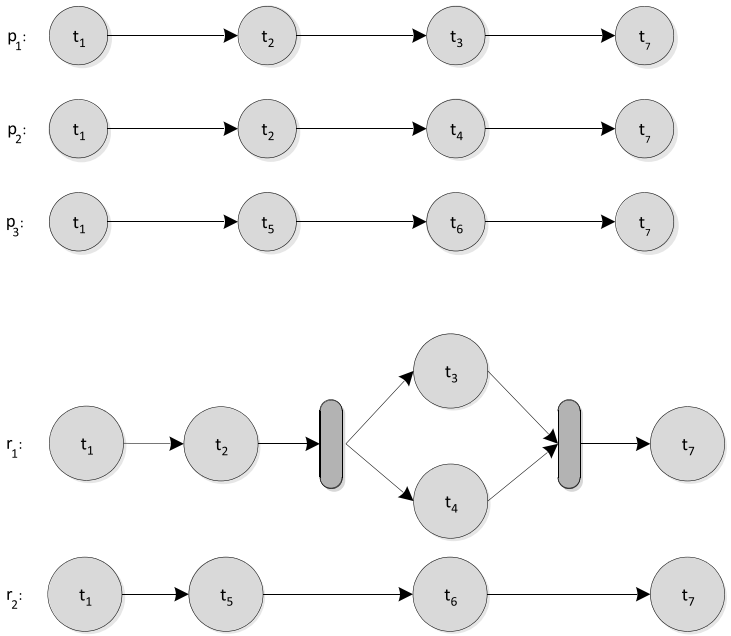

Fig. 2 Execution routes and paths of a process model

2 Multi-criteria Service Selection Model

2.1 Service Composition Model

Services are well-defined self-contained modules that provide business functionality. A service is accessible using a public interface that is described in terms of a standard interface description language. A task represents an abstract functionality that can be provided by a service. The service class S_i includes services that fulfill the functional requirements to execute a task t_i.

If existing services are assembled to provide a specific functionality we call the resulting component a service composition. A service composition can be described in terms of a process model that consists of tasks. The execution flow of a process model is defined by the sequential execution of tasks, conditional branches, and parallelization. A conditional branch is evaluated during the execution of a service composition, and a certain branch is taken. It is assumed that the branching probability of each branch is known or an appropriate estimate is available. The branches of a parallelization are executed in parallel. The parallelized branches are only joined when the execution of each branch is completed. A process model that consists of seven tasks, two conditional branches, and a parallelization is shown in **Fig. 1**. Note that services of seven service classes are used to provide the requested functionality to execute the tasks $t_i, i = 1, \ldots, 7$ in **Fig. 1**.

The set of all tasks is denoted by T. The notation $t_i \rightarrow t_k$ is used if t_i is a preceding task of t_k. The set $\hat{T} \subseteq T$ is the set of tasks without any preceding task, while $\tilde{T} \subseteq T$ is the set of tasks without any succeeding task. According to Yu et al. (2007) the execution structure of a service composition is described in terms of execution paths P and execution routes R. An execution path $p \in P$ is a path from $t_i \in \hat{T}$ to $t_k \in \tilde{T}$ that contains only one branch of each conditional branch and one branch of each parallelization. An execution route $r \in R$ is a sub-model of the process model that contains only one branch of each conditional branch but all branches of a parallelization. The execution probability $\nu(r)$ of $r \in R$ is calculated as the product of the branching probabilities of the conditional branches included in r. The notations $t_i \in T[r]$ and $t_i \in T[p]$ are used when a task t_i is part of the execution route r or the execution path p, respectively. The execution paths $P = \{p_1, p_2, p_3\}$ and execution routes $R = \{r_1, r_2\}$ of the service composition from **Fig. 1** are shown in **Fig. 2**.

The set of all services is denoted with $S = \bigcup_{t_i \in T} S_i$. A service binding b determines the service $s_{ij} \in S_i$ that is used to execute the task t_i. Hence, a service binding is a mapping:

$$b : T \to S, \qquad t_i \mapsto s_{ij} \in S_i. \tag{1}$$

In the remainder of this article, the service $b(t_i)$ that is bound to t_i is abbreviated as b_i. The set of all service bindings is defined as $B := \{b | b : T \to S, t_i \mapsto b_i \in S_i\}$.

The cost associated with a service invocation of $s \in S$ is denoted with $c(s) \in IR^+$. The availability of $s \in S$ is represented by $a(s) \in [0, 1]$. The actual response time of s is $r(s) \in IR^+$. Note that $r(s)$ is only known after the processing of s is completed. Hence, an estimate of the response time $\tilde{r}(s) \in IR^+$ is considered by an ex-ante performed service selection. A reasonable estimate can be obtained based on historical information provided by a monitoring system as proposed by Qiang et al. (2012). An empirical distribution of the response time of $s_{ij} \in S_i$ is represented by $K \in IN^+$ classes. The left endpoint of the k-th class is $r_{ij}^{k-1}$and the right endpoint is r_{ij}^k. The minimal response time is given by r_{ij}^0, while the maximal response time is r_{ij}^K. The remaining $k - 1$ values r_{ij}^k are chosen equidistantly between r_{ij}^0 and r_{ij}^K. The quantity h_{ij}^k is the number of service invocations observed for s_{ij} with $r_{ij}^{k-1} \leq r(s_{ij}) < r_{ij}^k$.

Following Yu et al. (2007), the expected cost $c(b)$, the availability $a(b)$, and the execution time $e(b)$ of a service composition with respect to the service binding b are calculated as:

$$c(b) = \sum_{r \in R} \nu(r) \cdot \sum_{t_i \in T[r]} c(b_i), \tag{2}$$

$$a(b) = \min_{r \in R} \prod_{t_i \in T[r]} a(b_i), \tag{3}$$

$$e(b) = \max_{p \in P} e_p(b). \tag{4}$$

In (4), $e_p(b)$ is the execution time of a path $p \in P$. Since the tasks of an execution path are sequentially executed, the execution time is calculated as:

$$e_p(b) = \sum_{t_i \in T[p]} \tilde{r}(b_i). \tag{5}$$

The execution time of a service composition must not exceed $\bar{e}$. If the execution time restriction is violated, the execution of a service composition is regarded to be failed. Facing with uncertain response times, the reliability $\psi(b)$ represents the probability that the execution time restriction will be fulfilled when the service composition is executed according to b. The minimal reliability that has to be achieved by a service binding b is ψ_{min}, i.e., b is feasible only if $\psi(b) \geq \psi_{min}$ holds.

The goal of the service selection is to determine tradeoffs between the conflicting cost minimization and availability maximization objectives. The problem is called the multi-criteria stochastic service selection (MCSS). The MCSS problem can be tackled by standard solution approaches when the objectives are combined into the integrated objective function:

$$I(b) = a(b) + w \cdot c(b). \tag{6}$$

The quantity w in (6) is the weight parameter used to balance the importance of the cost minimization objective against the objective to maximize the availability. The value of w depends on the scaling of the cost and the availability and the preferences of the decision maker. Combining the objectives is called an a-priori approach since the value of w has to be determined prior to the optimization without knowledge about possible alternatives.

A multi-criteria service selection avoids the parameter w to be determined prior to optimization. Availability maximization and cost minimization are conflicting objectives. Hence, each service binding represents a certain tradeoff between them and consequently, an unambiguous solution cannot exist. A feasible service binding b is called non-dominated when no other feasible service binding b' exists with $c(b') \leq c(b)$ and $a(b') \geq a(b)$, and at least one of these inequalities is strict. The entire set of non-dominated solutions for a problem instance is called the Pareto frontier.

2.2 Prospective Application Examples

In this subsection, we discuss some prospective applications of the techniques proposed in this paper because we are not aware of real-world applications of QoS-aware service-selection methods. Web Services are used to automate business processes. The QoS-aware service selection is typically considered in the context of composing Web Services to implement complex business functionality. The dynamic adaption of service compositions based on Web Services is supported by, e.g., the eFlow environment proposed by Casati et al. (2000). However, this subsection shows that the notion of a QoS-aware service selection can be easily extended to more general orchestration and deployment scenarios.

Viswanadham and Kameshwaran (2009) consider the orchestration of a network of activities in a value chain. An orchestrator is assumed which does not own capacities but has access to a large pool of service providers that can perform various activities in a value chain. Viswanadham and Kameshwaran (2009) report that the Hong Kong-based trading company Li & Fung collaborates with thousands of service providers. These providers are selected on short notice according to the activities required to fulfill an order. The selection is driven by several conflicting attributes such as the capacities of the service providers, the lead time, and the production cost. Although a mixed integer programming (MIP) solution approach based on SAW is presented, a heuristic multi-criteria decision support method will be beneficial for the orchestration problem to efficiently obtain possible alternative implementations in the value chain.

Another application of the QoS-aware service selection can be found in the area of service deployment in data centers. The services a business process is composed of are implemented in software components as, e.g., Enterprise Resource and Planning (ERP) systems, standard IT services, such as Directory Services, or as individually implemented Web Services (cf. **Fig. 3**). The existing software components can be deployed on internal servers or with a cloud infrastructure used as an Infrastructure As A Service (IAAS) like Amazon's Elastic Computing Cloud (EC2). We assume that QoS estimates are available for each option, obtained by, e.g., designed experiments to evaluate a system as proposed by Jamoussi et al. (2010). A deployment plan has to be determined for each software component in order to meet the QoS requirements of the business process and optimize QoS attributes.

An example process that consists of four tasks is shown in **Fig. 3**. The tasks are executed by services provided by packaged software like the Lightweight Directory Access Protocol (LDAP) and ERP systems and Web Services which can be hosted on servers inside the enterprise

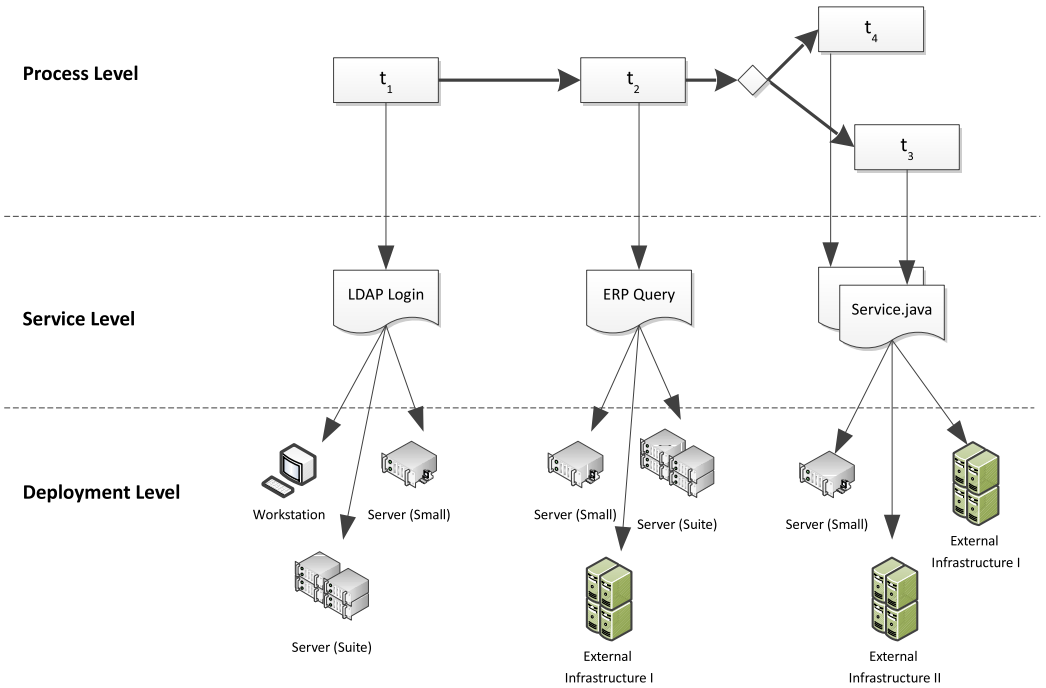

Fig. 3 Deployment alternatives of services used to implement a business process

or at IAAS. The costs of a deployment plan consists of the rental fees for the IAAS and the operating costs of the internal servers. In addition to the costs, the execution time of a business process and its throughput can be considered as conflicting QoS objectives.

The last example deals with future logistics. In the physical internet, physical objects are packaged in so called π-containers that are shipped in a multi-segment manner (Montreuil 2011). The segments are connected by π-hubs that allow for an efficient unloading/loading of π-containers. The transportation requestor has to select appropriate transportation providers to ship their goods from segment to segment to reach the final destination. The services, offered by the providers, are differentiated in terms of transportation time, transportation cost, and other quality measures. These services form a service class for each segment. Considering the transport on a segment as a task, a QoS-aware service selection allows for an identification of providers for each segment, ensuring that QoS goals such as minimizing of the total cost are fulfilled and total traveling time constraints are not violated.

2.3 Related Literature

A service selection model for service compositions with sequentially executed tasks that accounts for globally constrained QoS attributes is presented by Yu and Lin (2004). SAW is used to combine the QoS attributes to be optimized into an integrated objective function. The service selection is modeled as a combinatorial optimization problem and as a graph-based problem. The optimization problem is a special case of the NP-hard multiple choice knapsack problem (MCKP). Several exact solution approaches are studied for both the combinatorial and graph-based formulation with respect to their computational performance. The algorithm proposed by Pisinger (1995) outperforms the remaining approaches considered in Yu and Lin (2004) with respect to computing time.

Aggregation functions are used to estimate the QoS of a service composition that consists of conditional branches and parallelization. Jaeger et al. (2004) and Cardoso et al. (2004) propose a set of aggregation patterns that are used to define appropriate aggregation functions. Aggregation functions to estimate costs, to calculate the availability, and to aggregate the total execution time of a service composition are provided. Yu et al. (2007) propose a QoS-aware service selection model that relies on the QoS aggregation functions proposed by Jaeger et al. (2004). Multiple QoS attributes to be optimized are combined into an integrated objective function utilizing SAW. Several exact solution approaches are studied.

The computational burden to obtain an optimal service binding increases exponentially with an increasing number of tasks. Therefore, research effort was spent to develop efficient heuristic solution approaches to address the increasing computational complexity of large-scale service compositions. QoS-aware service selection approaches based on genetic algorithms (GAs) are developed by Jaeger and Mühl (2007) as well as Canfora et al. (2005b) independently from each other. It is shown that a service binding that is feasible with respect to given QoS constraints can be determined efficiently. However, the optimization of QoS attributes is not well addressed.

The uncertainty of QoS attributes is neglected by the service selection models discussed so far. Canfora et al. (2005a) consider uncertain QoS values and their impact on QoS requirements. They aim to determine the tasks of the process model, i.e., workflow slices that are not executed so far. A new service selection is performed for the identified workflow slices if the deviation exceeds a predefined threshold. This approach is refined by Ramacher and Mönch (2013) by focusing on uncertain response times and a globally constrained execution time.

The aforementioned service selection models are solely devoted to optimize a single objective function. In the case of multiple QoS attributes to be optimized, an integrated objective function

is derived by SAW. However, determining the weights required for an integration of the objectives is a non-trivial task especially in absence of further information, e.g., on the range of alternative solutions. Therefore, service selection models that are based on the principles of multi-criteria optimization are desired.

Liu et al. (2005) consider a NSGA-II approach for a multi-criteria service selection. However, they neither provide a problem encoding nor were computational experiments performed. A further approach that utilizes a GA to solve the service selection problem in a multi-criteria sense is proposed by Wada et al. (2011). Facing a cost minimization and throughput maximization objective, the determination of the set of non-dominated service bindings is addressed. A canonical GA is tailored to determine a set of non-dominated service bindings. A fitness function, similar to the one used by NSGA-II, evaluates a chromosome with respect to its domination rank and its distance to other solutions of the same population and thus fosters the generation of a non-dominated solution.

A multi-criteria optimization based on integer programming (IP) is proposed by Wiese et al. (2008). The service selection model considers uncertain response times and uncertain costs. The goal of the service selection is to minimize the Average Value at Risk of costs and the duration of a service composition while the availability of a service composition is globally constrained. The objectives are optimized independently from each other so that a set of non-dominated solutions is obtained.

With the rare exception of Wiese et al. (2008), multi-criteria service selection approaches do not account for uncertain QoS attributes. Considering the uncertainty of response times, the service selection aims to identify a robust service binding that allows a reliable execution of the service composition. Although other QoS attributes are considered, a similar goal is pursued by Wiese et al. (2008). However, as a solution approach they use IP which is not able to deal with large-scale service compositions in a reasonable amount of time because of the NP-hardness of the problem. Thus, the goal of the present article is to provide a heuristic solution approach that determines the Pareto frontier in a reasonable amount of time even for large-scale service compositions. Besides the multi-criteria optimization, the approach proposed in this article is the first approach that exploits problem specific heuristics of the service composition problem to improve the efficiency of the underlying metaheuristic.

3 Framework for Multi-criteria Service Selection

3.1 NSGA-II-Based Solution Approach

3.1.1 NSGA-II Principles

The NSGA-II scheme proposed by Deb et al. (2002) is a state-of-the-art metaheuristic used to determine an approximation of a Pareto frontier of combinatorial optimization problems. However, in principle, other metaheuristics are also possible as discussed by Talbi et al. (2012). NSGA-II relies on selection, crossover, and mutation to generate new populations of chromosomes. The selection mechanism is designed to foster the generation of Pareto optimal solutions. As part of the selection mechanism, a crowding operator improves the diversification of the solutions obtained for the Pareto frontier. A problem encoding and an appropriate assessment method have to be specified for the MCSS problem.

3.1.2 Problem Encoding and Genetic Operators

The MCSS problem allows for the following natural encoding scheme. A service binding for a service composition with n tasks is encoded by a chromosome that consists of n genes. The i-th gene identifies the service $s_{ij} \in S_i$ that is bound to task t_i. The i-th gene takes a value of the allele set $\{1, \ldots, |S_i|\}$.

Offspring chromosomes are determined using the uniform crossover operator. The offspring o_1 is derived from two parents c_1 and c_2 that are chosen by tournament selection from the current population. For each gene of o_1 a certain parent chromosome is selected from $\{c_1, c_2\}$ with a probability p_s. The value of the gene is set according to the value of the corresponding gene of the selected parent.

The mutation operator is used to randomly change the service binding of a chromosome selected with a certain mutation probability to avoid a premature convergence of the algorithm. The mutation operator selects the genes considered for mutation by chance. A gene is selected with a probability p_g. The value of the chosen gene is set to a randomly determined value of its allele set. The chromosomes of the initial population are determined by selecting at random the gene values from their allele sets.

3.1.3 Chromosome Assessment

The fitness of a chromosome c that encodes the service binding b^c is represented by the values $o_{cost}(c)$ and $o_{avail}(c)$. The expected cost $c(b^c)$ and the expected availability $a(b^c)$ are derived according to (2) and (3), respectively. In addition, the reliability $\psi(b^c)$ has to be considered to decide whether c is feasible or not.

A Monte Carlo Simulation (MCS) is used to estimate $\psi(b^c)$. The finite set of scenarios Ω is considered to sample the response times. A scenario $\omega \in \Omega$ represents the response time $r^{\omega}(s) \in IR^+$ for each $s \in S$ that is sampled from the response time distribution of s. The execution time of the service composition with respect to b^c and the scenario ω is $e(b^c, \omega)$. The value $e(b^c, \omega)$ is calculated according to (4) and (5) where $\tilde{r}(s) = r^{\omega}(s)$ is used. The reliability $\psi(b^c)$ is estimated by:

$$\psi^{MCS}(b^c) = \frac{\sum_{\omega \in \Omega} \kappa(\omega)}{|\Omega|}, \tag{7}$$

where $\kappa(\omega)$ is 1 if $e(b^c, \omega) \leq \bar{e}$ holds and 0 otherwise.

The chromosome c is infeasible if $\psi^{MCS}(b^c) < \psi_{min}$ is obtained. A common mechanism to deal with infeasible solutions is to introduce penalties. The penalty $P(c)$ of a chromosome c is defined as:

$$P(c) = \max\{\psi_{min} - \psi^{MCS}(b^c), 0\}. \tag{8}$$

Following Deb (2000), the fitness of a chromosome c is calculated as:

$$o_{cost}(c) = \begin{cases} c(b^c), \\ \quad \text{if no violation}, \\ c_{max} + P(c), \\ \quad \text{otherwise}, \end{cases} \tag{9}$$

$$o_{avail}(c) = \begin{cases} a(b^c), \\ \quad \text{if no violation}, \\ a_{min} - P(c), \\ \quad \text{otherwise}. \end{cases} \tag{10}$$

In (9) and (10), c_{max} and a_{min} are the largest cost value and the smallest availability value in the entire population, respectively.

Our tailoring of the NSGA-II approach that incorporates the encoding of the service binding and the corresponding evaluation method is called MCS-GA.

3.2 Heuristic Extension

A service binding encoded by a chromosome can be improved by replacing the services bound to tasks. The selection of appropriate services is driven by two heuristics which incorporate characteristics of the MCSS problem. As proposed by Ramacher and Mönch (2012), the heuristics are integrated into the metaheuristic to improve the performance of the algorithm.

3.2.1 MCSS Heuristics

The heuristics are related to a deterministic version of the MCSS problem. Therefore, the uncertain response time of a service $s \in S$ is represented by a suitable estimate $r^H(s) \in IR^+$. The estimate is derived based on the mean and a risk factor $\eta \geq 0$ dependent standard deviation term as:

$$r^H(s) = E[\mathrm{R}] + \eta\sqrt{Var[R]}, \tag{11}$$

where R is a random variable for the response time of s. The risk factor is adjusted during the optimization procedure to ensure feasibility for the stochastic problem.

Considering the response time estimates, a service binding b is feasible if $e(b) \leq \bar{e}$ holds and infeasible otherwise. Distinguishing these two cases, two heuristics are proposed:

- *Repair heuristic*: An infeasible service binding is adjusted such that the execution time of the service composition is decreased.
- *Improve heuristic*: A feasible service binding is adjusted such that the objectives are improved unless the execution time restriction is violated.

Each of the heuristics replaces a service b_i bound to t_i with a suitable service $s_{ij} \in S_i$. The replacement is carried out by the rebinding function RB that is defined as:

$$RB : B \times T \times S \to B, \qquad (b, t_i, s_{ij}) \mapsto b', \tag{12}$$

where

$$b'(t_k) := \begin{cases} b_k, & \text{if } t_k \in T \backslash \{t_i\}, \\ s_{ij}, & \text{otherwise.} \end{cases} \tag{13}$$

The suitability of a service s_{ij} is a measure to identify the services that are considered to replace the service b_i. Considering the suitability of an element is common in optimization procedures that tackle the MCKP. For example, Pisinger (1995) calculates the likelihood of an element to be included in an optimal solution as the ratio of its utility and its weight. The suitability of a service is calculated depending on two weight factors $w_{cost}(b)$ and $w_{avail}(b)$ as:

$$\begin{aligned} \mu_b(s_{ij}) = \big[& w_{cost}(b)\big(c(s_{ij}) - c(b_i)\big) \\ & + w_{avail}(b)\big(a(b_i) - a(s_{ij})\big)\big] \\ & \times \big[r^H(b_i) - r^H(s_{ij})\big]^{-1}. \end{aligned} \tag{14}$$

The quantity $\mu_b(s_{ij})$ estimates to which extent the cost and the availability of a service composition are changed with respect to a time unit when the service s_{ij} is used to execute the task t_i instead of b_i.

The set D_i of dominating services for the task t_i is

$$\begin{aligned} D_i = \big\{ & s_{ij} \in S_i | c(s_{ij}) \leq c(b_i), \\ & a(s_{ij}) \geq a(b_i),\ r^H(s_{ij}) \leq r^H(b_i) \big\}. \end{aligned} \tag{15}$$

Each service $s_{ij} \in D_i$ can be used to replace b_i without increasing the execution time of the service composition and without deteriorating the values of the QoS attributes to be optimized.

3.2.2 Repair Heuristic

The goal of the repair heuristic is to replace a service b_i with a service $s_{ij} \in S_i$ that has a lower response time to decrease the total execution time of the service composition. The tasks contribute differently to the reduction of the execution time. The suitability of a task t_i is calculated as the task weight $\Delta_b^R(t_i)$ that is defined for the repair heuristic as:

$$\Delta_b^R(t_i) = \frac{\sum_{\{p \in P | t_i \in T[p]\}} \max\{e_b(p) - \bar{e}, 0\}}{\sum_{p \in P} \max\{e_b(p) - \bar{e}, 0\}}. \tag{16}$$

The task weight $\Delta_b^R(t_i)$ is well-defined in the case that at least one execution path exceeds the execution time restriction. A value of $\Delta_b^R(t_i) = 0$ is obtained if t_i only belongs to execution paths that do not violate the execution time restriction. Apparently, in this situation a task t_i cannot contribute to reduce the execution time. The value $\Delta_b^R(t_i)$ increases if t_i is part of one or more execution paths that violate the execution time restriction.

The repair heuristic consists of two phases. In the first phase, only the services in D_i are considered. For each task t_i, the service $s_{ij} \in D_i$ with the lowest response time is selected from D_i and is used to replace b_i where ties are randomly broken. If the service binding is still infeasible, the heuristic proceeds with the second phase. In the second phase, the repair heuristic considers the combined index $\mu_b^R(s_{ij}) = \mu_b(s_{ij})/\Delta_b^R(t_i)$ to calculate the suitability of a service s_{ij}. The value of $\mu_b^R(s_{ij})$ is positive and small for s_{ij} when the cost compared to b_i is only slightly increased, the availability is only slightly decreased, and t_i has a high contribution to reduce the total execution time. The second phase is implemented by the following steps:

- *Step 1*: Determine the execution paths that violate the execution time restriction, i.e.
  $P_b = \{p \in P | e_b(p) > \bar{e}\}$. Stop the heuristic if $P_b = \emptyset$ holds. In this case, the service binding is feasible.
- *Step 2*: Determine the services that reduce the execution time of the service composition, i.e.

$$S_b^R = \big\{ s_{ij} \mid r^H(s_{ij}) < r^H(b_i), t_i \in T[P_b] \big\}. \tag{17}$$

  Stop the heuristic if $S_b^R = \emptyset$ holds. In this case, no feasible solution exists.
- *Step 3*: Randomly select the service $s_{ij} \in S_b^R$ with the lowest value for $\mu_b^R(s_{ij})$ and derive $b' = RB(b, t_i, s_{ij})$.
- *Step 4*: Set $b = b'$ and proceed with Step 1.

Note that the calculation of the value $\mu_b^R(s_{ij})$ in Step 3 is possible for all $s_{ij} \in S_b^R$ because $r^H(s_{ij}) \neq r^H(b_i)$ is ensured and $\Delta_b^R(t_i) > 0$ holds because only tasks are considered that belong to an execution path that violates the execution time restriction.

3.2.3 Improve Heuristic

This heuristic aims to improve the cost and the availability for a feasible service binding. The improved heuristic again consists of two phases. In the first phase, the service $s_{ij} \in D_i$ with the lowest value

for $w_{cost}(b)c(s_{ij}) - w_{avail}(b)a(s_{ij})$ is selected and bound to t_i to improve the objectives. The total execution time cannot be increased by selecting a service from D_i and thus the service binding remains feasible after the first phase.

In the second phase, the positive difference (called slack) between the execution time of an execution path and the execution time restriction is exploited to further improve the objectives. Analogously to the repair heuristic, the suitability of a task to contribute to an improvement without violating the execution time restriction is considered. The weight of a task t_i is calculated with respect to the slack of the execution paths it belongs to:

$$\Delta_b^I(t_i) = \frac{\min_{\{p \in P | t_i \in T[p]\}} \max\{\bar{e} - e_b(p), 0\}}{\sum_{p \in P} \max\{\bar{e} - e_b(p), 0\}}. \quad (18)$$

The value $\Delta_b^I(t_i)$ is high for a task t_i that only belongs to execution paths with a high slack. The task weight is 0 for a task that belongs to at least one execution path whose execution time is greater than or equal to $\bar{e}$. The task weight is considered in conjunction with μ_b in the combined index $\mu_b^I(s_{ij}) = \mu_b(s_{ij}) \cdot \Delta_b^I(t_i)$. The value $\mu_b^I(s_{ij})$ is higher for s_{ij} the more the cost is reduced and the availability is increased per time unit that the service composition is delayed by s_{ij}.

The improve heuristic consists of the following steps. The set $CS \subseteq S$ is considered as a set of services that are not considered by the heuristic anymore. Initially, $CS = \emptyset$ is used.

- *Step 1*: Determine the set of all services that improve the cost and the availability, i.e.

$$S_b^I = \{s_{ij} \mid c(s_{ij}) < c(b_i),\ a(s_{ij}) > a(b_i),\ s_{ij} \in S \backslash CS\}. \quad (19)$$

  Stop the heuristic if $S_b^I = \emptyset$ holds.
- *Step 2*: Randomly select $s_{ij} \in S_b^I$ with the largest $\mu_b^I(s_{ij})$ value and derive the binding

$$b' = RB(b, t_i, s_{ij}).$$

- *Step 3*: If $e(b') \leq \bar{e}$ holds set $b = b'$ and proceed with Step 1. Otherwise add s_{ij} to CS and proceed with Step 1.

Note that for each service $s_{ij} \in S_b^I$ the inequality $r^H(s_{ij}) > r^H(b_i)$ holds because of the first phase of the heuristic. Hence, the combined index $\mu_b^I(s_{ij})$ is well-defined for all $s_{ij} \in S_b^I$.

3.2.4 Hybridization of MCS-GA

The repair and improve heuristics are integrated into the MCS-GA to improve the service bindings encoded by the chromosomes. The integration determines the weight parameters $w_{cost}(b)$ and $w_{avail}(b)$, adjusts the risk factor η, and identifies the chromosomes on which the heuristics are applied.

Applying the heuristics on each chromosome is not appropriate since the heuristics implement a greedy search technique. Different solutions might be mapped to a single solution and thus, the diversification of a population is reduced. The chromosomes on which the heuristics are applied are selected according to a selection probability that is derived from a temperature parameter $\Theta \in [0, 1]$ and a characterization $\xi(P_i)$ of a population. The value of Θ is initially set to $\Theta = 1$. It linearly decreases after each generation such that $\Theta = 0$ is reached at the end of the algorithm. The population P_i is characterized with respect to the fraction of feasible service bindings:

$$\xi(P_i) = \frac{|\{b^c | c \in P_i, \psi^{MCS}(b^c) \geq \psi_{min}\}|}{|P_i|}. \quad (20)$$

The probabilities P_R and P_I of applying the repair and the improve heuristic, respectively, are set as:

$$P_R = \big(1 - \xi(P_i)\big)\Theta, \quad (21)$$

$$P_I = \xi(P_i)\Theta. \quad (22)$$

Two random numbers $r_R, r_I \in [0, 1]$ are determined independently after a chromosome c is generated. The heuristic h is applied on c if $r_h \leq P_h$ holds for $h = R$ or $h = I$. In a situation where many solutions are infeasible, the probability P_R is high. Hence, the repair heuristic is frequently applied and finally feasible solutions will be established in the population. On the other hand, when many solutions are feasible, the probability P_I is increased and positive slacks are exploited to improve the objectives.

The value of the risk factor η, initially set to 0, is adjusted by taking into account the feasibility of the solutions obtained by the heuristics. Each time a heuristic is applied on a chromosome c, the feasibility of c is evaluated according to the penalty term $P(c)$ defined by (8). Afterwards, η is adjusted by $\eta = \eta + P(c) \cdot \Theta$. The adjustment accounts for the feasibility of the MCSS problem because the risk factor η is increased relatively to the degree of violation of ψ_{min} that is represented by $P(c)$. The dependency on the temperature parameter ensures that the step size of the adjustment is decreased at the end of the algorithm to ensure the convergence of η.

The weights $w_{cost}(b)$ and $w_{avail}(b)$ are selected following Deb and Goel (2001) depending on a current service binding b as:

$$w_{cost}(b) = \frac{(c_{max} - c(b))}{(c_{max} - c_{min})^2}, \quad (23)$$

$$w_{avail}(b) = \frac{(a(b) - a_{min})}{(a_{max} - a_{min})^2}. \quad (24)$$

The values c_{max} and c_{min} are the maximal and minimal cost of a service binding in the current Pareto front. Analogously, a_{max} and a_{min} are the maximal and minimal availability. We assume $c_{max} \neq c_{min}$ and $a_{max} \neq a_{min}$. Otherwise, the weights are set to one. This weight selection forces the heuristics to favor the objective that is closer to its individual optimum. The hybridized version of MCS-GA is called MCS-GA-H.

3.3 Optimal Solution Approach

The ε-constraint method allows for an independent optimization of multiple objectives to obtain a Pareto frontier (cf. Ehrgott 2010 for more details). Instead of combining the objectives into an integrated objective function, a single objective is optimized in a certain step while the remaining objectives are transformed into constraints. An ε-constraint problem with K objectives $f_k,\ k = 1, \ldots, K$ to be minimized is given as:

$$\min f_k \quad (25)$$

subject to:

$$f_j \leq \varepsilon_j, \quad j = 1, \ldots, K,\ j \neq k. \quad (26)$$

The ε-constraint formulation for the MCSS problem is stated in (27)–(30). The quantities $E_{cost}, E_{avail} \in \{0, 1\}$ and $\varepsilon_{cost}, \varepsilon_{avail} \in IR$ are the parameters of the model. For $E_{cost} = 1$ and $E_{avail} = 0$

the model pursues the cost minimization whereas the availability is restricted to ε_{avail}. In the other case of $E_{cost} = 0$ and $E_{avail} = 1$, the model aims for the availability maximization where the cost is restricted to ε_{cost}. We obtain:

$$\min_{b \in B} E_{cost} c(b) - E_{avail} a(b) \tag{27}$$

subject to:

$$(1 - E_{cost}) c(b) \leq \varepsilon_{cost}, \tag{28}$$

$$a(b) \geq (1 - E_{avail}) \varepsilon_{avail}, \tag{29}$$

$$P\big(e(b) \leq \bar{e}\big) \geq \psi_{min}. \tag{30}$$

The inequalities (28) and (29) are the ε-constraints. The reliability constraint is formulated as a chance constraint through (30). The calculation of $a(b)$ can be linearized by a logarithmic transformation. The probability $P(e(b) \leq \bar{e})$ can be approximated by a scenario-based approach in which the response times used to calculate $e(b)$ are sampled from their response time distributions as proposed by Wiese et al. (2008). Hence, the model (27)–(30) can be implemented as a linear IP.

The model (27)–(30) is iteratively solved to obtain the set of Pareto optimal service bindings. The first iteration is started with $E_{cost} = 0$, $E_{avail} = 1$, $\varepsilon_{avail} = 0$ and $\varepsilon_{cost} = M$ where M is the sum of the cost of all services in S. The solution is the service binding with the availability a_{cur} where the cost is restricted to M. Afterwards, the IP is solved a second time with $E_{cost} = 1$, $E_{avail} = 0$, $\varepsilon_{avail} = a_{cur}$, and $\varepsilon_{cost} = M$, leading to the Pareto optimal service binding with the availability a_{cur} and the cost c_{cur}. The next iteration is started with $E_{cost} = 0$, $E_{avail} = 1$, $\varepsilon_{avail} = 0$, and $\varepsilon_{cost} = c_{cur} - \delta$ where δ is a value smaller than the cost difference of two arbitrary services in S. The procedure is repeated until the IP becomes infeasible for the parameters ε_{avail} and ε_{cost}. The generation of the problem instances tackled by the ε-constraint method ensures that no two services exists with the same cost such that $\delta > 0$ is always achieved. However, in a practical setting, $\delta = 0$ can occur leading to a situation in which the ε-constraint method is not applicable.

4 Computational Experiments

The performance of the MCS-GA and MCS-GA-H is evaluated with respect to the solution quality and the computational effort compared to the ε-constraint method presented in Sect. 3.3. Randomly generated problem instances are used. Although the problem instances are synthetic, they are generated according to a design of experiments that captures a large range of factors.

Table 1 Design of experiments

Factor	Design	Level	Count
Composition type (CTyp)	SMALL	Small, medium	2
	LARGE	Large	1
Number of services (m)	SMALL	4, 6, 8	3
	LARGE	10, 20, 30	3
Reliability restriction (ψ_{min})	SMALL	0.94	1
	LARGE	0.92, 0.94, 0.96, 0.98	4
Variance factor of response time (m_r)	SMALL	0.25	1
	LARGE	0.15, 0.25, 0.35	3
Execution time restriction (m_e)	SMALL	0.20	1
	LARGE	0.10, 0.15, 0.20, 0.25, 0.30	5

4.1 Design of Experiments

The factors for the problem instance generation are summarized in **Table 1**. Two designs, namely SMALL and LARGE, are distinguished. The factor CTyp identifies the process model of the service composition. The process model shown in **Fig. 1** is referred to as the medium process model. The small (large) process model is derived from the medium model by removing (adding) tasks. The small process model consists of five tasks whereas the large process model includes 25 tasks. Overall, 186 factor combinations are considered. For each of them, three independent problem instances are generated.

For each service class, m services are generated. The cost and the availability of a service $s_{ij} \in S_i$ are determined as $c(s_{ij}) = 100(1.5 - r_1)$ and $a(s_{ij}) = 0.9 + 0.1 r_2$ where r_1, r_2 are realizations of the uniformly distributed random variables $R_1, R_2 \sim U[0, 1]$.

The response time of s_{ij} is represented by an empirical distribution that consists of $K = 10$ classes. The minimal and maximal response time of $s_{ij} \in S_i$ is set to $r_{ij}^0 = 100(1 - m_r)\sqrt{r_1 r_2}$ and $r_{ij}^K = 100(1 + m_r)\sqrt{r_1 r_2}$. The frequency of each class is selected as $h_{ij}^k = 100 r_k$ where r_k is a realization of an uniformly distributed random variable $R_k \sim U[0, 1]$ for each $k = 1, \ldots, K$. The execution time restriction is set to $\bar{e} = (1 - m_e)\underline{r} + m_e \bar{r}$, where $\underline{r}$ and $\bar{r}$ are the minimal and maximal execution time of the service composition according to the response times of the generated services.

4.2 Results

The ε-constraint method is implemented using the solver LPSolve 5.5. The MCS-GA and MCS-GA-H are implemented using the C++ programming language upon the NSGA-II implementation of the MOMHLib class library. The experiments were conducted on an Intel Pentium IV CPU with 3.6 GHz and 4 GB RAM. The Pareto front determined by an approach $A \in \{\text{MCS-GA}, \text{MCS-GA-H}, \varepsilon\}$ is denoted by y_A. MCS-GA and MCS-GA-H are performed for five independent runs. A Pareto front is evaluated against a near-to optimal Pareto front denoted with y_{true} that is derived as the union of the non-dominated solutions of all Pareto frontiers obtained for a problem instance. The quality of a Pareto front y_A is evaluated using the following measures:

$$\text{ONVG}(y_A) = |y_A|, \tag{31}$$

$$\text{ONTVG}(y_A) = \big|\{b | b \in y_A \cap y_{true}\}\big|, \tag{32}$$

$$I_\varepsilon(y_A) = \max_{b \in y_{true}} \min_{b' \in y_A} \max\left\{\frac{c(b')}{c(b)}, \frac{a(b)}{a(b')}\right\}, \tag{33}$$

$$error(y_A) = \text{ONTVG}(y_A)/\text{ONVG}(y_A), \tag{34}$$

$$dist(y_A) = \frac{1}{\text{ONVG}(y_A)} \sum_{b \in y_A} \min_{b' \in y_{true}} d(b, b'), \tag{35}$$

Table 2 Computational results for the SMALL design

Factor	m	*error*		*dist*		ONVG		I_ε		Time (s)
		MCS-GA	MCS-GA-H	MCS-GA	MCS-GA-H	MCS-GA	MCS-GA-H	MCS-GA	MCS-GA-H	
Small	4	0.000	0.000	0.000	0.000	6.660	6.660	1.000	1.000	1
	6	0.000	0.000	0.000	0.000	4.870	4.870	1.000	1.000	4
	8	0.018	0.012	0.051	0.029	15.030	15.710	1.001	1.001	25
Medium	4	0.000	0.000	0.000	0.000	10.330	10.330	1.000	1.000	1265
	6	0.134	0.072	0.914	0.121	75.930	73.040	1.014	1.005	3688
	8	0.279	0.174	7.603	1.517	122.730	81.530	1.207	1.137	35362

Table 3 Computational results for the LARGE design

Factor		*error*		*dist*		ONVG		I_ε	
		MCS-GA	MCS-GA-H	MCS-GA	MCS-GA-H	MCS-GA	MCS-GA-H	MCS-GA	MCS-GA-H
m	10	0.635	0.593	1.141	0.561	162.670	156.150	1.127	1.007
	20	0.651	0.589	2.176	0.325	163.780	165.920	1.183	1.007
	30	0.611	0.601	3.212	0.354	157.120	172.180	1.206	1.005
ψ_{min}	0.92	0.636	0.596	1.354	0.402	161.290	166.270	1.170	1.005
	0.94	0.627	0.591	2.749	0.428	172.810	181.330	1.173	1.006
	0.96	0.641	0.594	1.699	0.407	158.150	156.440	1.167	1.006
	0.98	0.625	0.597	2.692	0.405	152.520	154.960	1.166	1.006
m_r	0.15	0.637	0.595	1.876	0.242	169.390	175.880	1.172	1.006
	0.25	0.644	0.582	1.899	0.423	139.920	135.930	1.169	1.006
	0.35	0.613	0.607	2.728	0.645	174.270	182.450	1.164	1.006
m_e	0.10	–	0.541	–	0.059	0.000	17.680	–	1.007
	0.15	0.670	0.555	9.893	0.088	24.020	61.900	1.158	1.007
	0.20	0.672	0.582	4.238	0.335	125.740	138.770	1.170	1.007
	0.25	0.632	0.628	1.036	0.560	262.700	256.210	1.171	1.005
	0.30	0.598	0.617	0.319	0.705	393.500	349.200	1.168	1.005

where

$$d^2(b,b') = \frac{(c(b)-c(b'))^2}{(c_{max}(y_{true})-c_{min}(y_{true}))^2} + \frac{(a(b)-a(b'))^2}{(a_{max}(y_{true})-a_{min}(y_{true}))^2}. \quad (36)$$

ONVG measures the size of y_A while ONTVG is the number of non-dominated solutions in y_A. The *error* measure evaluates the ratio of dominated solutions and Pareto optimal solutions in y_A. The multiplicative ε-measure I_ε determines the factor by which the front y_A is worse than y_{true} with respect to all objectives (Zitzler et al. 2003). Finally, $d(b,b')$ measures the distance between two solutions b, b'. This distance is used by *dist* that evaluates the average distance between y_A and y_{true}.

The parameterization of the MCS-GA and MCS-GA-H are determined based on some preliminary experiments. The population sizes {50, 75, 100, 150} and the mutation probability {0.1, 0.2, 0.3} were tested by trial runs. A population size of 100 and a mutation probability of 0.1 turned out to be the most efficient configuration of the algorithms in the majority of the considered trial runs. In addition, we also performed some experiments to determine appropriate values for p_s within the uniform crossover and for p_g in the mutation operator. We select $p_g = 0.5$ and $p_s = 0.5$, i.e., tossing a biased coin is not necessary, based on a trial and error strategy. Moreover, trial runs are used to determine the number of scenarios considered by MCS-GA and MCS-GA-H to estimate $\psi^{MCS}(b)$. It turns out that 100 scenarios are sufficient to reduce the error $|\psi^{MCS}(b) - \psi(b)|$ to less than 0.002.

The problem instances of the SMALL design are solved by the MCS-GA, MCS-GA-H, and the ε-constraint method. MCS-GA and MCS-GA-H are restricted to 600 seconds of computing time per problem instance. The results are shown in **Table 2**. The results are average values obtained for the problem instances with the characteristic stated in the factor column.

The problem instances of the LARGE design are only solved by MCS-GA and MCS-GA-H because of the increasing computational burden of the ε-constraint method. The results are shown in **Table 3** as average values in the same way as in **Table 2**. Again, the computing time per problem instance is 600 seconds.

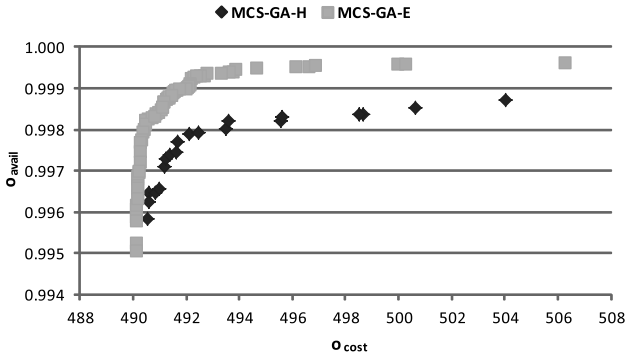

Fig. 4 Pareto frontier for a single problem instance

4.3 Discussion of the Results

The results obtained for the SMALL design show that both the MCS-GA and the MCS-GA-H are able to identify the true Pareto frontier for $m = 4$ and $m = 6$. The values for *error*, I_ε, and *dist* increase only slightly when the search space is enlarged by considering more than four services for each task for the medium process model. However, the measures indicate that the Pareto frontiers obtained for all problem instances are close to the true Pareto frontier generated by the ε-constraint method. The MCS-GA-H always leads to better results than the MCS-GA with regard to the *error* and *dist* measure.

The advantage of MCS-GA-H compared to MCS-GA increases with a larger search space. MCS-GA is almost always outperformed by MCS-GA-H with regard to experiments performed according to the LARGE design. It turns out that the MCS-GA-H is in particular beneficial if the response time restriction is severe. For the experiments with $m_e = 0.1$ the MCS-GA is not able to identify any feasible solution.

The size of the search space is influenced by m. The values for *error*, I_ε, and *dist* obtained for MCS-GA increase, revealing that the approximation of y_{true} deteriorates with an increasing value of m. The uncertainty of the response times determined by m_r turns out to have only a minor impact on the results.

The results clearly show that both algorithms described in this article are applicable to obtain appropriate approximations of a true Pareto frontier where the computational burden is significantly decreased compared to an optimal solution approach. However, if the response time restrictions are severe, heuristic extensions are required to obtain a feasible service binding. In almost all other cases, the performance of the MCS-GA is improved by the heuristic extensions.

The value of the robust service selection is pointed out in **Fig. 4**. A problem instance for the medium-type process model is solved by the MCS-GA-H and a MCS-GA version that only considers the expected values of the response times (MCS-GA-E). It turns out that the Pareto front obtained by MCS-GA-H is very close to the frontier of MCS-GA-E. Hence a low price is paid for ensuring robustness with respect to feasibility since the objective function values deteriorate only slightly compared to the model with mean response times. Evaluating the solutions of MCS-GA-E by a simulation of 100.000 requests, however, shows that none of the solutions obtained by MCS-GA-E fulfill the reliability constraint whereas the reliability constraint is satisfied by the MCS-GA-H solutions in 98 % of all solutions.

In addition, five problem instances are solved by MCS-GA-H and MCS-GA-E for each type of process model, and the ratio of the solutions that fulfill the reliability constraint is calculated. We find that only 10 %, 4 %, and 1 % of the solutions determined by MCS-GA-E satisfy the reliability constraint for the small, medium, and large process model respectively. In contrast, 98 %, 95 %, and 93 % of the solutions computed by MCS-GA-H fulfill the reliability constraint.

5 Conclusions and Future Research

A QoS-aware service selection involves multiple, usually conflicting and possibly uncertain QoS attributes. Implementing a service composition based upon existing services requires a tradeoff between the conflicting objectives. Hence, exposing the set of possible alternative implementations supports a decision maker in finding such a tradeoff. This article studies a multi-criteria QoS-aware service selection problem with uncertain response times. The considered service selection accounts for a robust service selection that ensures a reliable execution. A solution approach to determine the Pareto frontier of alternative service bindings based on the NSGA-II metaheuristic is proposed. The metaheuristic is extended by heuristics that exploit particular characteristics of the service selection problem. The computational experiments demonstrate that our solution approach is suited for an operational service selection because near Pareto optimal solutions are obtained in a few minutes. In addition, we show that it is worthwhile to anticipate uncertainty when QoS-aware service selection decisions are made. This feature of our approach is important from a managerial point of view. We believe that this will allow for applications in value chains, supply chain management, and multi-segment intermodal transportation. In addition, we are confident that the combination of improved data availability and algorithmic advances will lead to new types of decision support systems.

Although the proposed service selection model aims for a robust service selection with respect to uncertain QoS attributes, future research will be directed to extend the proposed multi-criteria service selection model to account for service failures and unavailable services. Necessary adjustments of the service compositions have to be identified to ensure a feasible service selection also in the case of a volatile environment in which service failures have to be considered.

References

Aier S, Bucher T, Winter R (2011) Kritische Erfolgsfaktoren für die Gestaltung serviceorientierter Informationssysteme: Ableitung und empirische Evaluation eines Kausalmodells. WIRTSCHAFTSINFORMATIK 53(2):75–87

Bertsimas D, Sim M (2004) The price of robustness. Operations Research 52(1):35–53

Bichler M, Lin KJ (2006) Service-oriented computing. IEEE Computer 39(3):99–101

Canfora G, Di Penta M, Esposito R, Villani ML (2005a) QoS-aware replanning of composite web services. In: Proc of the IEEE international conference on web services, Orlando, pp 121–129

Canfora G, Di Penta M, Esposito R, Villani ML (2005b) An approach for QoS-aware service composition based on genetic algorithms. In: Proc of the international conference on genetic and evolutionary computation, Washington, pp 1069–1075
Cardoso J, Sheth AP, Miller JA, Arnold J, Kochut K (2004) Quality of service for workflows and web service processes. Journal on Web Semantics 1(3):281–308
Casati F, Ilnicki S, Jin L, Krishnamoorthy V, Shan M-C (2000) Adaptive and dynamic service composition in eFlow. In: Proc of the 12th international conference on advanced information systems engineering, Stockholm, pp 13–31
Deb K (2000) An efficient constraint handling method for genetic algorithms. Computer Methods in Applied Mechanics and Engineering 186(2):311–338
Deb K, Goel T (2001) A hybrid multi-objective evolutionary approach to engineering shape design. In: Proc of the first international conference on evolutionary multi-criterion optimization, Zürich, pp 385–399
Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2):182–197
Eder J, Panagos E, Pozewaunig H, Rabinovich M (1999) Time management in workflow systems. In: Proc of the 3rd international conference on information systems, Poznan, pp 265–280
Ehrgott M (2010) Multicriteria optimization, 2nd edn. Springer, Berlin
Jaeger MC, Rojec-Goldmann G, Mühl G (2004) QoS aggregation for web service composition using workflow patterns. In: Proc of the 8th international enterprise distributed object computing conference, Monterey, pp 149–159
Jaeger M, Mühl G (2007) QoS-based selection of services: the implementation of a genetic algorithm. In: Proc of the KiVS workshop 2007: service-oriented architectures und service oriented computing, Bern, pp 359–370
Jamoussi Y, Driss M, Jèzèquel J-M, Ben Ghèzala HH (2010) QoS assurance for service-based applications using discrete-event simulation. International Journal of Computer Sciences 7(6):1–11
Liu S, Liu Y, Jing N, Tang G, Yu T (2005) A dynamic web service selection strategy with QoS global optimization based on multi-objective genetic algorithm. In: Proc of the 4th international conference on grid and cooperative computing, Beijing, pp 84–89
Montreuil B (2011) Toward a physical Internet: meeting the global logistics sustainability grand challenge. Logistics Research 3:71–87
Papazoglou MP, Traverso P, Dustar S, Leymann F (2008) Service-oriented computing: a research roadmap. International Journal of Cooperative Information Systems 17(2):223–255
Pisinger D (1995) A minimal algorithm for the multiple-choice knapsack problem. European Journal of Operational Research 83(2):94–410
Qiang H, Jun H, Yun Y, Schneider JG, Hai J, Versteeg S (2012) Probabilistic critical path identification for cost-effective monitoring of service-based systems. In: Proc of the 9th international conference on services computing, Hawaii, pp 178–185
Ramacher R, Mönch L (2012) Heuristiken zur multikriteriellen Komposition von Diensten in dienstbasierten Informationssystemen. In: Proc of the Multikonferenz Wirtschaftsinformatik 2012, Braunschweig, pp 1171–1182
Ramacher R, Mönch L (2013) Reliable service reconfiguration for time-critical service compositions. In: Proc of the 10th international conference on services computing, Santa Clara, pp 184–191
Scholl A (2001) Robuste Planung und Optimierung. Grundlagen, Konzepte und Methoden, Experimentelle Untersuchungen. Physica-Verlag, Heidelberg
Talbi E-G, Basseur M, Nebro AJ, Alba E (2012) Multi-objective optimization using metaheuristics: non-standard algorithms. International Transactions in Operational Research 19:283–305
Viswanadham N, Kameshwaran S (2009) Orchestrating a network of activities in the value chain. In: Proc of the 5th annual IEEE conference on automation science and engineering, Bangalore, pp 501–506
Wada H, Suzuki J, Yamano Y, Oba K (2011) E3: multi-objective genetic algorithms for SLA-aware service deployment optimization problem. IEEE Transactions on Services Computing 99(12):1155–1156
Wiese W, Hochreiter R, Kuhn D (2008) A stochastic programming approach for QoS-aware service composition. In: Proc of 8th IEEE international symposium on cluster computing and the grid, Lyon, pp 226–233
Yu T, Lin K-J (2004) Service selection algorithms for web services with end-to-end QoS constraints. In: Proc of the IEEE international conference on e-commerce technology, San Diego, pp 129–136
Yu T, Zhang Y, Lin K-J (2007) Efficient algorithms for Web services selection with end-to-end QoS constraints. ACM Transactions on the Web 1(1):6
Zitzler E, Thiele L, Laumanns M, Fonseca CM, da Fonseca VG (2003) Performance assessment of multiobjective optimizers: an analysis and review. IEEE Transactions on Evolutionary Computation 7(1):117–132

Abstract

René Ramacher, Lars Mönch

Robust Multi-criteria Service Composition in Information Systems

Service compositions are used to implement business processes in a variety of application domains. A quality of service (QoS)-aware selection of the service to be composed involves multiple, usually conflicting and possibly uncertain QoS attributes. A multi-criteria solution approach is desired to generate a set of alternative service selections. In addition, the uncertainty of QoS-attributes is neglected in existing solution approaches. Hence, the need for service reconfigurations is imposed to avoid the violation of QoS restrictions. The researched problem is NP-hard. This article presents a heuristic multi-criteria service selection approach that is designed to determine a Pareto frontier of alternative service selections in a reasonable amount of time. Taking into account the uncertainty of response times, the obtained service selections are robust with respect to the constrained execution time. The proposed solution approach is based on the Non-dominated Sorting Genetic Algorithm (NSGA)-II extended by heuristics that exploit problem specific characteristics of the QoS-aware service selection. The applicability of the solution approach is demonstrated by a simulation study.

Keywords: Service composition, QoS-aware service selection, Genetic algorithm, Uncertain QoS