Chapter 13
Service Selection in Web Service Composition: A Comparative Review of Existing Approaches

Mahboobeh Moghaddam and Joseph G. Davis

Abstract Web service composition (WSC) offers a range of solutions for rapid creation of complex applications in advanced service-oriented systems by facilitating the composition of already existing concrete web services. One critical challenge in WSC is the dynamic selection of concrete services to be bound to the abstract composite service. In this paper, we provide a comprehensive review of the existing proposals for service selection, and a comparative analysis of the optimization and automated negotiation-based approaches.

13.1 Introduction

Service-Oriented Computing (SOC) has emerged as an important computing paradigm in recent years. Its main feature is that it utilizes one or more interoperable services (software components which implement specific functionalities) as fundamental building blocks to support rapid, low-cost development of distributed applications in heterogeneous environments [23]. SOC represents a new generation of distributed computing platforms which builds on past distributed computing approaches. It is distinguished by the addition of new design layers, governance considerations, and a set of preferred implementation technologies [18].

M. Moghaddam (✉) · J. G. Davis
School of Information Technologies, University of Sydney,
Sydney, NSW 2006, Australia
e-mail: mahboobe@it.usyd.edu.au

M. Moghaddam
National ICT Australia (NICTA), Australian Technology Park,
Eveleigh, NSW 2015, Australia

J.G. Davis
e-mail: joseph.davis@sydney.edu.au

DOI: 10.1007/978-1-4614-7518-7_13,

More recently, web services have been advanced as the technology of choice for realizing service oriented computing and its associated set of strategic goals. Web services are self-contained, modular business applications with open, Internet-oriented, standards-based interfaces [52]. The communication between web services is via standards-based technologies which give users the opportunity to access different web services, independent of their hardware, operating system, or even programming environment. This supports organizations with a technology to create services which can be easily discovered and consumed by external users.

One of the critical research challenges in realizing the vision of agile and collaborative software development using web services is *Web Service Composition* (WSC) which involves creating a composite service by combining different web services to provide a new value added service [5]. *Service selection*, as the problem of selecting the most appropriate web services from the pool of available ones that best match the functional and non-functional requirements and constraints specified by the requester has been researched extensively. The primary goal of this paper is to provide a comprehensive review of a range of proposals for service selection in WSC. A comparative analysis of the two dominant approaches based on optimization and automated negotiation is also included.

The remainder of this paper is organized as follows: Sect. 13.2 introduces web service composition lifecycle to provide a common ground on WSC definition. A brief overview of service discovery approaches is presented in Sect. 13.3. In Sect. 13.4, we discuss the main challenges involved in arriving at a service selection solution. A comprehensive review of the existing service selection approaches, optimization-based, negotiation-based, and the hybrid approaches is presented in Sect. 13.5. Section 13.6 includes a comparative analysis of the two main approaches. The paper concludes with a brief summary and an outlook for future research in Sect. 13.7.

13.2 Web Service Composition

Although a single web service has its own value for its users, the functionality offered by the individual web services is limited. The true potential of web services can only be realized through assembling multiple web services into more powerful applications with more sophisticated functionalities; i.e. *Web Service Composition* (WSC). Manual composition of web services is time consuming, error-prone, generally hard and not scalable [5]. Hence, a family of approaches to web service composition aims to fully or partially automate the composition process.

The two main streams in the automatic WSC approaches are: *workflow-based* approaches and *AI planning-based* approaches [5, 47]. Workflow-based approaches are inspired by the similarity of an abstract composite service to an abstract business process, and the similarity of a concrete composite service to a running workflow system. This similarity has helped web service research community to build on the accumulated knowledge that has emerged in the workflow and business process

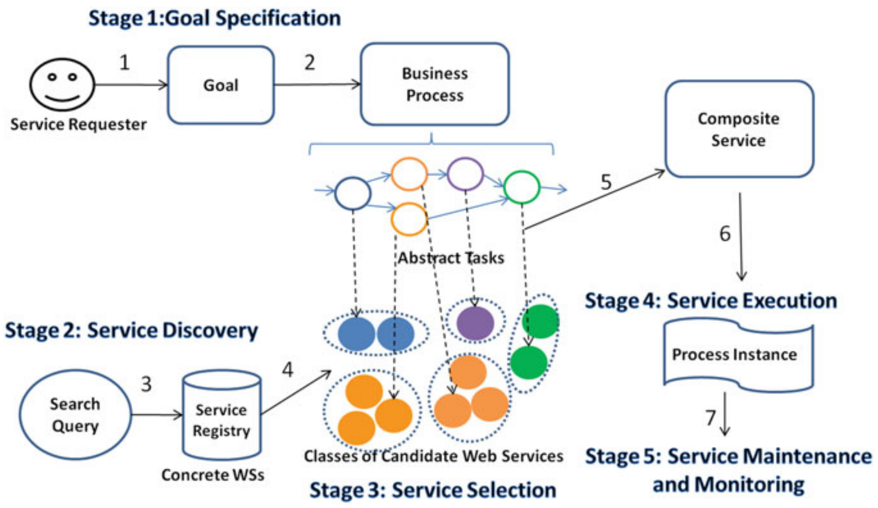
Stage 1:Goal Specification
1
2
Service Requester
Goal
Business Process
Composite Service
5
6
Abstract Tasks
Stage 4: Service Execution
Process Instance
7
Stage 2: Service Discovery
Search Query
3
Service Registry
4
Concrete WSs
Classes of Candidate Web Services
Stage 3: Service Selection
Stage 5: Service Maintenance and Monitoring

Fig. 13.1 Web service composition lifecycle

design research area [5]. We will discuss this approach in more detail by developing a model of Web Service Composition Lifecycle (Fig. 13.1).

In the AI planning-based approaches, service composition is viewed as a planning problem. Generally, a planning problem has the following elements: description of the *initial state* of the world, definition of the *desired goal state*(s), and a description of the *set of possible actions* which can transform world's state from one to another. The planner agent aims at finding the sequence of actions that will transform the world's state from the initial state to the goal state. In the WSC realm, a composite service is represented as a goal state to be achieved and the available web services are represented as the set of actions that can transform the world (or agent's) states. At the end of a successful planning exercise, a plan is generated. This plan constitutes the chosen web services and the order of their execution in such a way that they, in combination, will deliver the required functionality.

The main difference between the two approaches is that AI planning-based techniques generally do not make any assumption relating an abstract view of the composite service.[1] In contrast, this abstract view is a key element in the workflow-based approaches. In this paper, our focus is on the workflow-based approaches as the research in the business process and workflow-related areas have proven applications in industry, which has, in turn, helped to improve the academic research.

The lifecycle of a typical workflow-based WSC solution is illustrated in Fig. 13.1. In this lifecycle, the first stage is *the goal specification*, where the service requester's goal and preferences are defined. Following this, the goal is decomposed (semi)automatically into an abstract business process (BP) comprising a set of tasks,

[1] There are exceptions to this generalization, such as the work of McIlraith and Son [39]. They have a similar concept to an abstract business process, refered to as the high-level generic procedure.

each with clear functionality, along with the control and data flow among them. The Quality of Service (QoS) requirements for the BP (end-to-end quality) as well as for each participating task are also specified.

During the next stage, *service discovery*, concrete web services that match the tasks' functional and non-functional requirements are located by searching a service registry that holds information about available concrete web services. Service discovery is performed to find a match for each of the participating tasks. At this stage, it is very likely that more than one candidate will be found for each task that, while satisfying the basic required functionality, may be offered with different QoS attributes values, i.e. different levels of availability, price, etc.

Service selection is the stage following service discovery. At this stage, a variety of techniques is proposed by the research community that helps the service requester to select the web services that best match the specified requirements of the individual tasks and the business process. After finding the matches for all tasks and binding each task to its chosen web service, the *concrete composite service* is created. During service execution stage, a process instance is created by executing the composite service. The process instance would be continuously monitored for further responses toward any failure or change in its status at the final stage of WSC, i.e. *service maintenance and monitoring*.

Service discovery and selection, as two fundamental steps in the web service composition lifecycle, have been a major focus of service-oriented computing research. Before any interaction happens between service requester and provider, the former needs to locate the web service that best matches her task description. Researchers have mainly focused on service discovery techniques for a single task. In contrast, service selection has been addressed at two levels: for a single task, and for the complex BP. This second level is specifically required during WSC and the complexity level is exacerbated by the fact that service selection is performed for a set of dependent tasks. Hence, many researchers have considered service discovery and service selection as two separate stages to break down the complexity.

Following this approach, service discovery and service selection are treated as two distinct stages in our discussion of WSC lifecycle (Fig. 13.1), where the output of the former is the input to the latter stage. Even though our primary focus is service selection, we present a brief overview of the approaches to service discovery in the next section, before discussing a range of service selection approaches. This will help clarify the expected output of service discovery which is used as the input to service selection.

13.3 Service Discovery Approaches

In WSC, service discovery (also referred to as matchmaking) is the process of finding a concrete service match for each task in the BP. WSC solutions need to define the precise and specific *search criteria* to be executed against the *web service registry* to find the match for each task.

What to include in this search criteria varies among different solutions. The matchmaking algorithm proposed in [15] ranks functionally equivalent services on the basis of their ability to fulfill the service requester's functional and non-functional requirements while maintaining the price below a specified budget. The proposed matchmaking framework in [40] is based on the web service context where context is defined as all the information needed for enabling interactions between the service requester and providers. Semantic and behavioural information are used in [8, 9] for service matchmaking during WSC. Web service behaviour is the order of execution of the service operations or the order of message exchange with a service and the constraints governing the operations' execution sequence [17]. The selection algorithm proposed in [17] takes into account not only the functional requirements but also the transactional properties, and QoS characteristics of web services. Transactional properties guarantee consistent outcome and correct execution of the composite service. An information retrieval approach is suggested in [26] for discovering and ranking web services automatically, given a textual description of the desired services.

The variation in what to include in the search criteria arises from the fact that a web service can be defined from different perpectives, including: *functionality, QoS attributes, interface, semantics, behaviour,* and *context*. Web service interface is an essential aspect in the web service description (W3C working group note on Web Service Architecture [7]). However, what additional aspects to be included in the service definition is largely dependent on the application domain specificities. Service discovery proposals, including the ones mentioned above, have each added one or more aspects to the web service description, in addition to the interface. Elements of service specification affect the criteria included in the search request from the demand side, and the type of information to be stored in the service registry from the supply side.

Based on the search query criteria, service registry returns a number of candidate services. At this point, WSC enters the next stage, namely composite service selection. The input to the composite service selection stage is a set of *classes* of services. Each class contains services that can perform the same functionality, but they may differ on other aspects, such as QoS attributes.

13.4 Service Selection Challenges

In this section, we discuss the main challenges involved in the service selection problem. These challenges are: the NP-hardness of this problem for a composite service and the resulting scalability concern, the need to distinguish the abstract business process from its possible set of execution paths, defining the aggregation functions for the QoS attributes, and elicitation of the service user's preferences about different QoS attributes for the trade-off analysis of the candidate services.

NP-Hardness and Scalability

Composite service selection can be modelled as a multi-dimension multi-choice knapsack problem (MMKP), which is known to be an NP-hard problem in the strong sense [46]. This means that for large problems, it is unlikely that an optimal solution can be found given a reasonable amount of computational effort. Hence, there is a need for heuristic approaches when the problem size is too large to be solved by optimization procedures [46]. A number of heuristic algorithms for the service selection problem have been proposed in the literature; good exemplars include [6, 41, 60]. Some researchers have proposed a Genetic Algorithm approach to solve the scalability problem [10, 29, 37]. An alternative proposal to reduce the computational time of the service selection search algorithm is to shrink the search space. For instance, Alrifai et al. [2] has proposed pruning the service candidates that are not likely to be part of the optimal solution, by computing the service skyline for each service class.[2]

From Business Process to Execution Path

The assumption in workflow-based service composition approaches is that the required composite service is described at an abstract level as a high-level business process [4]. The business process is a collection of generic service tasks with defined control-flow and data-flow dependencies among them. Different languages and models have been used for describing the composite service, or more precisely its equivalent business process, such as UML activity diagram [4], statechart [62], extended BPEL [1], or YAWL [17].

Regardless of the modelling notation, different control-flow constructs are allowed in the existing process modelling languages such as *sequence, loop, parallel execution,* and *conditional branching*. Some control structures such as loop and conditional branching need special consideration. For these, the runtime structure is different from the abstract structure. For instance, only one of the tasks in the conditional branching would be selected for execution. This means that a BP might be executed along different paths, based on the control-flow at runtime. Each possible path of BP execution is called *an execution path*. During service selection, *an execution plan* is created by assigning web services to the tasks of an execution path.

Researchers have used different techniques to translate a BP to its corresponding execution paths, such as *loop peeling* [4], or *loop unfolding* [60, 62] to treat loop structures. In the former approach, every loop is annotated with the expected maxi-

[2] For a set of d-dimensional data points, the skyline is a subset of the points where no point in it is dominated by any other member. If $\overrightarrow{p}(p_1, ..., p_d)$ and $\overrightarrow{q}(q_1, ..., q_d)$ are two points in the d-dimensional data set, p dominates q *iff* $\forall i \in [1, d]$, $p_i \succeq q_i$ and $\exists j \in [1, d]$, $p_j \succ q_j$ [59]. The notation $\succeq$ is defined as being *better than or equal*, and $\succ$ as *better* than. In the service domain, a *service skyline* is the set of providers where no provider is dominated by any other, in terms of the offered values for QoS attributes.

mum number of its iterations, considering a probability distribution for the number of loop iterations. In the latter case, the loop is unfolded by cloning the functions in the loop for a number of times such as the maximal loop count, which can be obtained from process execution history or the process designer.

Aggregation Functions

A critical challenge in service selection is how to measure the end-to-end quality of the composite service. The aggregated value of a QoS attribute should take into account the QoS attribute value of the individual services participating in the composite service, and the business process structure. For example, the overall price of a composite service can be defined as the sum of the prices of all the participating services. However, for execution time, we need a more complex aggregation function, e.g. one that returns the maximum execution time among the parallel services, adds up the execution times of sequential services, and combines these two values if there are both parallel and sequential structures in the BP.

In [30] and its extension [31], Jaeger et al. have proposed aggregation functions for some QoS attributes such as execution time, cost and throughput, supporting a comprehensive set of structural patterns that can be found in workflows. Zeng et al. [62] has proposed aggregation functions for attributes such as execution price, execution duration, and reputation, supporting basic workflow patterns such as loop, sequence, conditional branching, and concurrent threads. Other aggregation functions have been also developed [4, 10, 48, 62].

Defining the Weights of QoS Attributes

There is a general assumption in the literature that the service requester has a clear idea of the importance of a QoS attribute which let her assign a scalar weight to each QoS criterion. But this may not be realistic, especially as the number of QoS attributes involved in the selection criteria increases. Some researchers have challenged this assumption. Wang [54] has proposed a resolution process for determining the linguistic weights of QoS criteria based on a group of participants' preferences. Yu and Bouguettaya [59] has proposed two algorithms for calculating the service skyline. Determining the skyline of a set of data requires pair-wise comparison of all the members of the data set which can be very expensive in terms of computational time and memory usage. The proposed algorithms in [59] exploit the indices of the service operations to compute the skyline more efficiently. The computed skylines guarantee the inclusion of the best user desired service providers without any user intervention.

13.5 Service Selection Spectrum

The input to the service selection stage is a set of classes of web services. The candidate web services in one class provide the same functionality, but they may vary according to other aspects such as QoS attribute values. QoS attributes (or non-functional properties) are the constraints defined over service functionality [45]. They can be categorized as:

- Technical domain-independent attributes,[3] such as: *response time, availability, reliability, robustness* (the ability of the service to continue its work in the presence of invalid, incomplete or conflicting inputs),
- Non-technical domain-independent attributes, such as: *execution price, penalty, discount, reputation*,
- Domain-dependent attributes which are only meaningful in a specific application domain, such as: *refresh time* for a traffic monitoring service [15].

Service QoS profile, provider's offered values for service QoS attributes, plays a central role in service selection research. Different providers may offer the same service at different levels of quality to maintain their competitive advantage over each other [40]. As well, a single provider might offer the same functionality with ranging quality levels to cover a wider range of customers. Moreover, at the composite service level, the QoS of the final composite service is the key factor to ensure service requester's satisfaction [62].

We have surveyed the range of service selection approaches discussed in the literature, based on the underlying assumptions regarding the QoS profile. The variation in the assumptions is illustrated as a spectrum in Fig. 13.2. Corresponding to the two extremes of the spectrum are the two important trends in the service selection literature: *Optimization-based* approaches which typically assume a predetermined not-customizable QoS profiles and, *negotiation-based* approaches which permit QoS profiles to be flexible and negotiable. In the following sections, we provide a

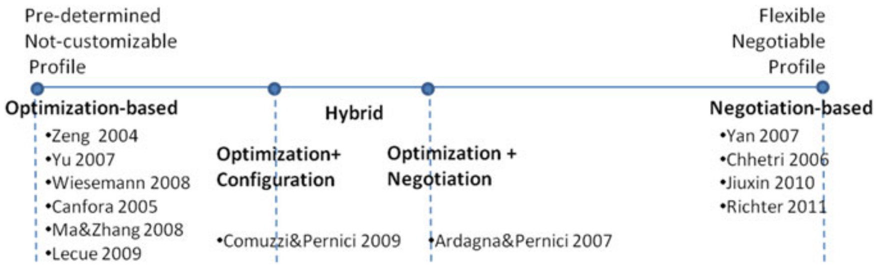

Fig. 13.2 Service selection spectrum based on QoS profile assumptions

[3] Due to space limitations, attributes such as privacy, security, and trust are not included in this paper.

comprehensive review of the important contributions on *service selection for a composite service* (or composite service selection).

13.5.1 Optimization-Based Approaches

Service selection can be modelled as an optimization problem. The optimization approach has appeared under different names such as *QoS-driven* or *QoS-aware web service composition, web service composition optimization,* and *optimum concretization*. Optimization can be performed at two levels: *local optimization* for an individual task, approaches such as [1, 17, 44] and *global optimization* for the BP, followed by for example [4, 60, 62].

Local Optimization

At the local level, the best service for individual tasks is chosen, one task at a time, regardless of the task dependencies with other tasks in the BP or the end-to-end quality requirements of the composite service. In this approach, services are ranked based on some criteria, including all service QoS attributes. The dominant technique to rank services is to assign a score to each web service, using utility theory. In utility theory (from microeconomics), the service requester or provider preferences can be mapped to values of utility, where higher utility means greater preferences [56]. To avoid the complexities of multi-dimensional utility function elicitation, each QoS attribute and the price have an independent utility function, based on the assumption of the independence of the outcomes of utility functions originating from *Multi-attribute Utility Theory (MAUT)* [34].

The offered value for the jth QoS attribute, q_j, ($j \in J$: set of all QoS attributes), by web service s, is mapped to a value between 0 and 1 using a single attribute linear utility function, denoted as U_j in Eq. (13.1). In this equation, q_j^{max} and q_j^{min} are the maximum and minimum values offered for q_j by all the candidate web services of the same functionality class.

$$U_j(q_j) = \begin{cases} \frac{q_j - q_j^{min}}{q_j^{max} - q_j^{min}} & \text{if larger } q_j \text{ more desirable} \\ \frac{q_j^{max} - q_j}{q_j^{max} - q_j^{min}} & \text{if smaller } q_j \text{ more desirable} \end{cases} \tag{13.1}$$

To get the aggregated utility of all the QoS attributes offered by service s (denoted as $U(s)$ in Eq. (13.2)), the weighted sum of the individual utility functions is calculated using a normalized weight (w_j) for each attribute specifying its importance. The sum of the normalized weights assigned to different QoS attributes (by service requester) should add up to 1, Eq. (13.3).

$$U(s) = \sum_{j \in J} w_j \times U_j(q_j) \tag{13.2}$$

$$\sum_{j \in J} w_j = 1 \tag{13.3}$$

Global Optimization

Even though the local optimization approach optimizes local service selection, it may not lead to a global optimality for the end-to-end QoS of the BP. Besides, it is not possible to set global constraints for the composite service in the local approach. To overcome these limitations, global optimization approaches have been proposed. In one such approach, optimization is carried out for the overall BP, and the requester can define end-to-end requirements and constraints for the overall BP. It is still possible to have a local selection strategy, beside the global optimization process, to address service requester's concerns about individual task quality. This can be achieved by applying the local QoS constraints as filters to the list of the candidate services returned by the service registry. The realization of the optimization problem's elements for composite service selection is as follows:

- Objective Function: The general objective function illustrated by research community for service selection is to maximize service requester's *satisfaction* from the execution of the composite service. To measure such satisfaction, researchers again draw on utility theory. The objective function is constructed by the weighted sum of the end-to-end QoS attributes' utility functions. An example objective function is presented in Eq. (13.4) below, where the service requester wants to minimize the price (or maximize the price utility, U_P) of the composite service, at the same time, maximizing the availability's utility, U_A.

$$\text{Maximize} \qquad w_P \cdot U_P + w_A \cdot U_A \tag{13.4}$$

$$\text{Subject to:} \qquad U_P = \sum_i \frac{P^{MAX} - \sum_j p_{ij} \cdot x_{ij}}{P^{MAX} - P^{MIN}} \tag{13.5}$$

$$U_A = \sum_i \frac{\prod_j a_{ij} \cdot x_{ij} - A^{MIN}}{A^{MAX} - A^{MIN}} \tag{13.6}$$

$$\sum_{i=1}^{m_j} x_{ij} = 1, \qquad x_{ij} \in 0, 1 \tag{13.7}$$

The price offered by service$_{ij}$ for executing task$_j$ in the BP is denoted as p_{ij}, the aggregation of all the offeres for the tasks in the BP as $\sum_j p_{ij} \cdot x_{ij}$, and the maximum and minimum price offers for the business process as P^{MAX} and P^{MIN}. A similar explanation applies to Eq. (13.6) for availability. Given a BP with J number of tasks, there will be J classes of candidate services where all the m_j candidate services in the jth class, can execute jth task ($j \in J$). Then, the decision variable x_{ij} is defined to be equal to 1 if the candidate web service$_{ij}$, in service class j, is assigned to execute task j or zero otherwise. Equation (13.7) ensures that only one service is selected from each class to execute the related task.

- Constraints: Service requester may have some constraints over the value of the QoS attributes which affects the choice of services for the BP. For example, a maximum budget is available to get the composite service, or the execution time might not exceed a maximum for the composite service to be useful.
- Decision Variables: The choice of what will represent the decision variables determines the type of optimization problem. The dominant approaches are modelling the problem as *Integer Linear Programming (ILP), Genetic Algorithm (GA), Constraint Satisfaction,* and *Stochastic Programming*.

Integer Linear Programming (ILP)

One way of solving this optimization problem is to model it as an integer linear programming problem [4, 62]. In the ILP approach, decision variables are integers representing whether a particular service is selected for executing a specific task or not, similar to x_{ij} in Eq. (13.7).

This approach helps researchers utilize any of the many available ILP solvers today. However, these solvers are effective when the size of the problem is small. An increase in the number of candidate web services leads to the increase of the number of decision variables, which in turn results in the explosion of the search space, and the number of the conditions to be checked. Thus, the ILP approach is limited by how large the BP (number of the tasks) is and how many candidate services exist. Another limitation of the linear programming approach is that both the objective function and the constraints should be *linear*, regardless of the complexity of the QoS attributes and different structures that can be found in the BP.

Genetic Algorithm (GA)

To overcome the limitations of ILP approaches, researchers such as [10, 29, 37] have proposed to apply genetic algorithm for the service selection problem. Any GA starts with encoding the candidate solution in a computer processable manner, called *genome*. The GA creates a *population* of generally random solutions which will be evaluated according to a *fitness function*. Then, based on some selection criteria, some individuals are selected for reproduction. The motive to the reproduction is that the new generation will contain better solutions than the old one. Near optimal solutions can be found by repeating these steps. The algorithm stops when some conditions are met, e.g. a specific number of generations.

In the service selection domain, Canfora et al. [10] and Jaeger and Muehl [29] applied a simple one-dimensional coding schema for the problem representation, while others, including Ma and Zhang [37], have used more complex representations such as a relation matrix coding schema. In the former case, each *individual* represents the assignment of the candidates to the tasks. In the latter one, the matrix can represent all the execution paths of the BP at the same time. The GA's fitness function is designed to maximize some QoS attributes and minimize some others. When dealing with end-to-end QoS attributes, the aggregation function for each QoS attribute needs to be defined. Evidently, there is no need to build "linear" aggregation functions (in contrast to the ILP approach).

GA is an unconstrained search technique [21], making it necessary to find ways for integrating service selection constraints into the search process. The widely used technique is the *additive penalty method* where a penalty cost that is proportional to the total violation of each of the constraints is added to the fitness function [27]. Other techniques to incorporate constraints into the GA search, such as [11] and [42], have not found application in service selection literature.

Constraint Satisfaction

One variant of the optimization approach is presented by Lecue and Mehandjiev [36]. They argue that future semantic web will cater for millions of services, making *scalability* (in terms of the time required for WSC) the main objective to achieve. They proposed a fast selection approach which might not lead to an optimal composition. Modelling service selection as a constraint satisfaction problem, they use a stochastic search method (more precisely, a hill-climbing algorithm) to find the first set of services that satisfy the set of defined constrains (both in terms of functional and non-functional requirements).

Rosenberg et al. [50] has proposed to model the service selection problem as a *Constraint Optimization Problem (COP)*. COP is a generalization of the constraint satisfaction problem where constraints are weighted and the goal is to find a solution maximizing a function of weighted constraints. The main idea in their proposal is that instead of having all the constraints as hard ones, i.e. must be satisfied, there can be defined soft constraints which are optional and it would be "nice" to have them. This will lead to a more flexible composition process. Based on this idea, the proposed CO algorithm does not find the best solution that exists in the search space (in terms of the utility gained by service requester). Rather it searches for the best solution within the boundaries of constraints. They add up all soft constraints to form an objective function, trying to maximize it. However, as they have mentioned, this approach has scalability problem, not suitable for large problems.

Stochastic Programming

Stochastic Programming is another optimization approach to solve the service selection problem in the presence of uncertainty. For example Wiesemann et al. [55] argues that the nature of QoS attributes such as response time and price is *non-deterministic*, and hence the WSC should be treated as a decision problem under *uncertainty*. To incorporate the uncertainty, they assume that the decision maker (the service requester) uses a particular quantile-based risk measure called *average value-at-risk (AVaR)* to quantify the risks associated with time and cost uncertainties. In the optimization objective function, they minimize the AVaR of the random variables defined for the service response time, and invocation cost. More precisely, they build *the worst-case risk functions* corresponding to execution time and price, using the associated AVaR measures. These two criterion functions constitute the optimization's objective function. In their experiment, they compared their risk-aware formulation of WSC in terms of the execution time and the price of the resulted composite services, with those of the deterministic formulation of the problem. According to their findings, for every deterministic composite service, there exists a risk-aware composition with smaller cost and execution time.

13.5.2 Negotiation-Based Approaches

Negotiation-based approaches constitute an important stream of contributions to the composite service selection problem. *Negotiation* is a process of reaching an agreement that is beneficial to the involved parties through information exchange and compromises [35]. Negotiating parties usually have different preferences over the negotiation issues and they seek to reconcile these differences through negotiation.

In computer science and related research, negotiation as distributed search through a space of potential agreements [32], has been used for many years to solve a variety of problems, e.g. resource allocation in grid computing, and getting agents to cooperate or compete over a common goal in multi-agent systems. In the context of computer science research, we should make it clear that what we mean by negotiation here is *an automated process* where negotiation is performed automatically by a piece of software such as an agent, a web service, or a third-party broker system. The automated negotiator replaces the human negotiator and performs negotiation on negotiator's behalf.

In the web service domain, researchers have employed negotiation mainly for (semi)automatic creation of *Service Level Agreement (SLA)*. In general terms, SLA is an agreement between the service consumer and the provider. It may also be referred to as *contract, policy,* or *license*. In service oriented infrastructure, SLA is an automatically processable contract between a service and its client, where the client can be an organization, a person, or another service [53]. In *SLA negotiation*, service provider and requester negotiate over SLA terms such as QoS attributes, rewards, penalties, and deliverables, in order to come up with a formal SLA at the end of the process [63]. SLA negotiation solutions are divided by the assumption that the service provider is predetermined before the negotiation or not. The two corresponding approaches are called *pre-contractual SLA negotiation*, and *dynamic provider selection*.

- Per-contractual SLA Negotiation [25]: In this case, the negotiation is performed after service discovery and selection, meaning that the service provider is already determined. Service parameters are negotiated and fixed in order to define the concrete service which will be carried out. This is a one-to-one negotiation process between service requester and the selected service provider. Proposals in this area include, but not limited to [14, 24, 64].
- Dynamic Provider Selection: Here, the negotiation is performed after the service discovery and as a service selection mechanism, aiming at dynamically selecting the service provider that best matches the service requester's non-functional requirements. This is a one-to-many negotiation process between the service requester and the candidate service providers. A successful negotiation output can be used for contract specification.

Basically in the dynamic provider selection approach, a high-level negotiation process (overall negotiation process) is conceptualized that negotiates for the overall BP. It consists of multiple negotiation sub-processes (briefly negotiation process) each associated with one task in the BP. Each negotiation process, in turn, may include multiple negotiation threads, one thread for each candidate provider, to choose the best service for the specific task.

When building an automated negotiation solution, several key components comprising the general negotiation framework [19, 43] should be addressed. These critical components are:

1. Negotiation Object: the set of issues that the parties negotiate to reach an agreement over their values,
2. Negotiation Protocol: the communication and message exchange rules among negotiation parties,
3. Decision-making Model: the rules that the interacting parties follow to decide when to start negotiation, how to prepare an offer, acceptable agreement range, and the time to abandon negotiation.

Meanwhile, when dealing with negotiation at the BP level, negotiation by itself may not be enough for achieving the end-to-end QoS requirement and ensuring a successful overall negotiation. A further management layer, referred to in the literature as coordination [12], becomes necessary. In Fig. 13.3 below, we present a WSC negotiation framework. This extends the general negotiation framework with an additional component i.e. coordination model which includes aspects of coordination strategy and architecture as explained below:

Coordination Strategy: involves decisions on (a) time to initiate negotiation processes for each task: All parallel? Sequential? With what priority?, (b) the type of information to collect from ongoing negotiation processes and/or finished ones to improve the negotiation result, and (c) actions to take for improving the negotiation result or prevent its failure, based on the collected information.

Coordination Architecture: involves how many and what type of negotiators are involved in negotiation (agents, web services, broker systems), and the required number of coordination layers and their configuration.

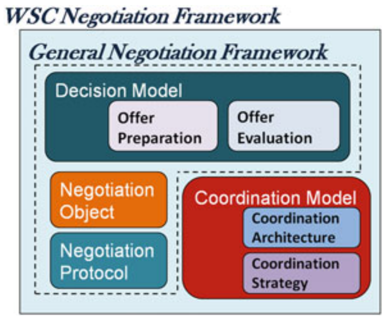

Fig. 13.3 WSC Negotiation framework adapted from the general negotiation framework [19, 43]

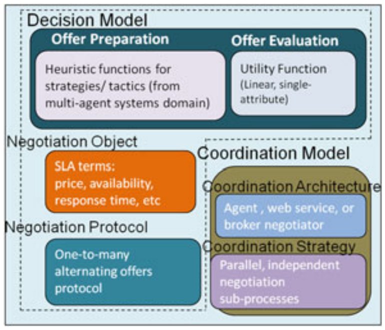
Decision Model
Offer Preparation
Offer Evaluation
Heuristic functions for strategies/ tactics (from multi-agent systems domain)
Utility Function (Linear, single-attribute)
Negotiation Object
SLA terms: price, availability, response time, etc
Coordination Model
Coordination Architecture
Agent , web service, or broker negotiator
Negotiation Protocol
One-to-many alternating offers protocol
Coordination Strategy
Parallel, independent negotiation sub-processes

Fig. 13.4 The realization of the WSC negotiation framework based on the current literature

We discuss below the realizations of the key elements of the framework in extant research. A summary of this discussion is included in Fig. 13.4.

Negotiation Object

In service selection, the negotiation object has already been fixed: the service requester negotiates over the value of QoS attributes with different service providers. QoS attributes can be negotiable or non-negotiable. Negotiable attributes are those whose values can be determined at run-time, during service invocation [15]. Negotiation is performed over a range of values for each term; i.e. the service requester and the provider each has a minimum and a maximum admissible value for a QoS attribute. Price, availability, and response time are the more commonly included terms in recent service selection experimental investigations [48, 58, 64]. When the negotiation object includes more than one issue (or attribute), the negotiator needs to know the relative importance of each issue. This is usually realized through a normalized weight for each negotiation issue.

Negotiation Protocol

Although an overall negotiation process is conceptualized for the composite service, the actual negotiation processes that ultimately occur are bilateral negotiation between service requester's and candidate provider's agents. Some researchers have used a general bilateral protocol [4, 14, 24, 49], also called as *bilateral message exchange* or *bargaining*. This general protocol consists of a series of message exchanges between the two parties in terms of *offers* and *counter-offers*, until one of them accepts an offer or withdraws from the negotiation, e.g. due to reaching the maximum negotiation time. Some researchers, including [12, 57, 64], have followed a standard protocol such as *FIPA ICN IP* [22]. This protocol allows multi-round bidding, supporting one-to-many negotiation. Under this protocol, the negotiation initiator issues the initial *call for proposals (CFP)*. The other parties in negotiation (contractors) answer by sending an offer or by refusing to participate in the negotiation. The initiator may accept or reject an offer or reply with a revised CFP. The negotiation terminates when the initiator accepts one or more offers, or refuses all the bids without issuing a new bid, or if all the contractors refuse to bid.

Some researcher proposes *generic* negotiation protocols. The idea is not to bind the negotiation solution to a particular protocol at design time. Rather, delaying the determination of the suitable negotiation protocol until the actual execution of the negotiation process to make a flexible solution. For example, by extending the current WS-Agreement specification [3], Hudert et al. [28] defines a separate stage for protocol determination, during which the negotiating parties agree on a common negotiation protocol before the actual negotiation process starts.

Decision Making Model

The two important parts of a negotiation decision model are: how to evaluate a received offer as to whether accept it or not (*utility function*) and how to prepare a counter-offer (*tactic/strategy*).

(a) Utility Function

Each negotiator needs to specify its preferences about the negotiation object. These preferences guide its decisions during negotiation. In service selection, the dominant approach to express them is through utility theory; referred to in the previous section. Many researchers addressing SLA negotiation have employed single attribute linear utility function to evaluate the value of an individual issue [4, 57, 64]. This utility function is similar to the Eq. (13.1) mentioned in the foregoing. Here, q_j^{max} and q_j^{min} are defined as the maximum and minimum admissible values for jth QoS attribute according to the negotiator's preferences and constraints. The *parametric* single attribute utility function [14], and *multi-attribute* utility function representing the relative preference with respect to each pair of attributes [24] have also been discussed in the literature.

As QoS profile typically involves more than one attribute, the more commonly used technique to measure the utility of a profile with multiple attributes is to assign a normalized weight to each attribute and calculate the overall utility using a weighted linear additive function; similar to the aforementioned Eqs. (13.2) and (13.3).

(b) Negotiation Tactics

The two main approaches discussed in the literature to generate a counter-offer are *concession* and *trade-off*. In the concessionary approach, with every new offer the negotiator concedes to the other side of negotiation (opponent) by preparing an offer that has a lower utility value for itself, and apparently a higher utility value for the opponent. How much concession to make, and the pace of offering concessions to progress the negotiation process are determined by the negotiation influential factors. *Time, resource,* and *opponent behaviour* are three factors proposed by Faratin et al. [19], leading to three families of tactics: *time-dependent, resource-dependent,* and *behaviour-dependent* (or *imitative*).

In contrast, a negotiator with the trade-off approach tries to keep its utility value stable at a desirable level (the *aspiration* level) throughout the negotiation, while generating an offer that has more utility value for the opponent. This can be achieved by trading-off between the values of different issues [20], i.e. lowering the values of some QoS attributes while demanding more on some others. Such a strategy maximizes the chance of the offer to be accepted. Considering the fact that the negotiator usually has no information about the opponent preferences and utility function, the main challenge is how to determine which offer increases the opponent's utility value. The trade-off strategy proposed by Faratin et al. [20] uses the concept of fuzzy similarity [61] to approximate the preferences of the opponent. Assuming that the opponent's last offer reflects its preferences, the negotiator uses it as a reference point and prepares a counter-offer that is most similar to it. In the Yan et al. [57] proposal, the authors take advantage of the one-to-many negotiations occurring for a composite service. The utility value of all the received offers is calculated, and the one with the best utility is used as a reference point for preparing the counter-offer.

Faratin et al. [19, 20] and Comuzzi et al. [13, 14] have proposed heuristic approaches to define the counter offer for bilateral negotiation. Faratin's heuristic functions [19] are widely adopted by researchers, e.g. by [4, 49, 64] due to the clear distinction of tactic families (based on time, resource, and opponent behaviour), the clear mathematical representation, and the analysis of negotiation convergence for different parameters of the model.

The tactics in the Faratin model consider the influential factors in any negotiation. However, negotiation in WSC is a special case of negotiation consisting of multiple one-to-many negotiation processes. This introduces the opportunity for considering other influential factors to define new tactics. For example the Global Negotiation States factor proposed in [33] reduces the need for unnecessary negotiations in a one-to-many negotiation. The received counter-offers are compared to each other, and if all the counter-offers are far from the initial offer, the negotiator should be ready to make bigger compromise. Otherwise, if any counter-offer is more desirable than the negotiator's own offer, negotiator will raise its expectations and prepares the next offer based on the value of this desirable counter-offer.

(c) Negotiation Strategy

Negotiation strategy is another part of the decision model. Conceptualized at a higher level of abstraction than the negotiation tactics, it aims to maximize the utility function of the negotiator for a contract [19], by determining when to use which tactic to prepare the counter-offer, or what combination of tactics to use. More precisely, strategy can be thought of as the pattern of change in the weight of different tactics over time [63]. Taking it to one step further, Di Nitto et al. [16] states that strategy is not just about how to weight different tactics over time, but it can also address the following factors: (a) Changing the importance of negotiation issues over time, e.g. prefer availability over the response time if the latter cannot be improved so far, (b) Changing the severity of the constraint, e.g. relaxing some constraints on the values of some negotiation issues to reflect more concession when the negotiation time is about to expire.

Deciding on the best strategy for a negotiator involves the challenges addressed mostly in game-theory, microeconomics, and multi-agent systems and is outside the scope of this paper.

Coordination Model

To avoid the complexity of dependent negotiation processes, researchers including [33, 48, 57] assume negotiation processes to be independent and concurrent. For the same reason, no information is collected during an ongoing negotiation process.

In the Yan et al. proposal [57], the coordinator takes part only at the end of the process to either confirm or reject the negotiation result. Extending [57], Richter et al. [48, 49] attempted to make the coordinator more actively involved in the negotiation. Thus the coordinator does not wait for all the negotiation processes to finish. Rather, when a negotiation process finishes successfully, the surplus of the negotiation issue is calculated. Surplus is the difference between the actual agreed value and the least desired value (e.g. maximum payable price from the service requester point of view) of the negotiation attribute. Subsequently, it is distributed over failed or unfinished negotiation processes of those tasks which have dependencies to the task producing the surplus. The dependency is determined based on the QoS attribute under negotiation, and the task's position in the process, and is maintained in a tree-format. However, redistributing surplus may prevent the failure of the negotiation process when service requester has severe QoS requirements. It is not helpful in situations where negotiation fails due to the limited negotiation time available.

13.5.3 Hybrid Approach

There are service selection approaches which are not based on pure optimization or negotiation. In this section, we summarize two of the more important contributions.

Optimization + Configuration Approach

One attempt to proceed from a totally predetermined QoS profile to a more flexible one is the work by Comuzzi and Pernici [15]. In their approach, rather than providing a single value for each QoS attribute, the service provider publishes the set of values that they can support for each QoS attribute. For example, a provider offering a Traffic Monitoring Service can publish the offered quality for the refresh time (the time interval between the updates of the traffic information) as $\{2\,h, 1.5\,h, 1\,h, 0.5\,h\}$. This means that refresh time can be offered with any of the intervals of 0.5, 1, 1.5, or 2 hours. Additionally, instead of assuming a single value for the price, they have proposed a pricing model, including a set of pricing functions for the QoS attributes. Each attribute's pricing function determines how much it will cost for the service requester to select a specific level of quality. The web service total price is calculated as the sum of its constituting pricing functions.

The proposed service selection technique is in fact a local optimization where the web service with the lowest price for the *minimum quality profile* is selected. Minimum quality profile of a service consists of the lowest level of quality for each QoS attribute which still satisfies service requester's quality demand. When service selection is completed, a subsequent *agreement configuration* step is performed. During the configuration step, the difference between the price of the low quality profile of the selected service and the service requester's budget is used to improve upon the offered service quality for requester.

This research does not assume a pre-determined QoS profile for neither the service offer nor the service request. Instead, the service provider is able to publish the quality profile in the form of different quality levels that he supports. Besides, a higher level of flexibility is supported for the service price offering with the proposed pricing model. Thus, service requester can receive a personally-configured service, based on her preferences and constraints. However, the flexibility of the QoS profile in negotiation-based approach does not exist here, as no negotiation actually takes place. Rather, a configuration process tailors the service quality based on the requester's preferences and budget.

Optimization + Negotiation Approach

The research by Ardagna and Pernici [4] is another attempt to relax the assumption about a fully pre-determined QoS profile to a more flexible one, by combining optimization with negotiation. They start service selection as a MILP optimization

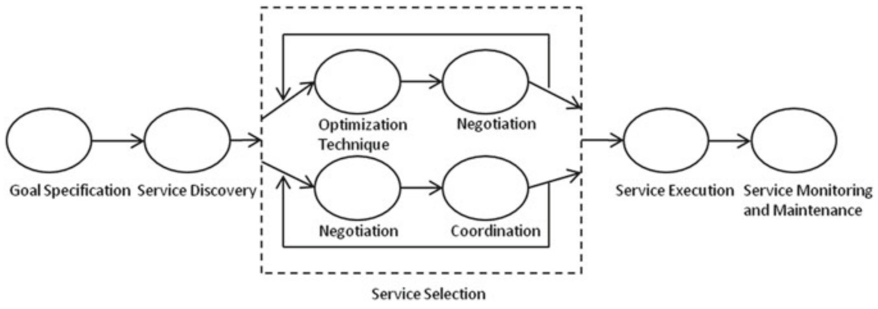

Fig. 13.5 Different perspectives on applying negotiation for service selection during WSC

problem. But if the optimization process fails to find a feasible solution due to sever QoS constraints for example, a negotiation process would initiate.

At the beginning of the negotiation process, first the execution plan that satisfies the maximum number of constraints is identified. Then, negotiation starts with any service provider that contributes to violating the global constraints of this execution plan. After the negotiation is completed, the providers who have agreed to improve their offered quality of service, in return for a higher price, will be added to the optimization space. In other words, negotiation is only used to find new QoS attribute values for web service invocations and it expands the optimization solution domain. As the last step of service selection, optimization is repeated with the new solution domain to find a feasible solution.

As mentioned in their paper, identifying the maximum number of constraints that can be satisfied is an NP-hard problem. Thus, they have assumed the global constraints are limited which allows to find the maximum number of violated constraints through an exhaustive search. Comparing their approach with pure negotiation-based approaches, coordination is not required here (Fig. 13.5). In fact, provider selection is performed through optimization, and not negotiation. However, in contrast to optimization-based approaches, the providers have a chance to improve their offered quality if their existing offers do not satisfy service requester's requirement.

13.6 Comparison

We present a comparison of the optimization and automated negotiation-based approaches to the service selection problem. The bases that we have used for the comparison are the following: the key elements of the proposed solutions, the nature of the QoS profiles in each approach, the reference disciplines that have influenced the literature, the ways in which the methods from the reference disciplines have been applied, the experimental strategies employed to validate the models, and some of the better-known tools.

The key elements in the optimization approach are a range of optimization models which specify one or more objective functions, a set of constraints and decision variables. For automated negotiation, the focus typically is on a set of negotiation objects or issues, a negotiation protocol, a complex decision model required for automated agents to conduct negotiation, and models for coordinating the negotiation process that unfolds over time and for arriving at a final decision on the concrete web services for the composite application.

The use of optimization techniques is generally restricted to situations in which the QoS profiles are pre-determined and fixed. This is somewhat unrealistic in light of the relatively dynamic environments that characterize the selection and composition of web services. Furthermore, it is not clear whether the web service providers will be willing to always create and publish static QoS profiles. The negotiation approach is suited for situations in which, at the very least, the price is negotiable. In many cases, price and at least a subset of the QoS attribute values are flexible which allow for more complex negotiation scenarios.

Automated negotiation approaches for service selection have drawn extensively on the more general agent-based negotiation literature for concepts, functions, and frameworks. These have the potential to address more realistic scenarios in the context of service selection. In particular, the relaxation of the assumption regarding static QoS profiles is an improvement over the less flexible optimization-based approaches. However, the dynamic aspects of negotiation approaches, including the need for a complex decision model and a cordination model, complicate the problem of finding globally optimum solutions. The hybrid models and proposals may offer potentially useful research directions in this regard.

The disciplinary influences that inform the former arise in mathematical and computing sciences whereas the negotiation approach draws on a broader, interdisciplinary body of knowledge. The contributing disciplines to negotiation approaches include computer science and artificial intelligence, behavioral sciences, economics and game theory, and mathematics, to name a few. Synthesizing robust negotiation models poses significant challenges and the results to-date on their applications in web service selection are still in their infancy. Much work still remains to be done on creating effective coordination models for automated negotiation to lead to useful outcomes for composite service selection.

The experiments that have been reported using both optimization and negotiation-based solutions cannot be described as convincing in that they typically incorporate very few QoS attributes to be considered representative of real-world scenarios. While the availability and maturity of available tools for optimization-based approach are very good, the tool space for negotiation is sparse. Though a range of optimization tools are widely available, scaling to larger problem size (in terms of number of tasks involved in BP, number of candidate services for each task, number of QoS attributes involved, number of global constraints etc), will often involve having to settle for feasible solutions, if any.

In general, optimization based approaches tend be more applicable under relatively static conditions in which the QoS profiles are fixed and the attribute values are pre-determined. However, if there is flexibility in the attribute values and tradeoffs

Table 13.1 The comparison of optimization- and negotiation-based approaches

Comparison bases	Optimization-based	Automated negotiation-based
Basic elements of the solution	Objective function, decision variables, constraints	Negotiation object (issues), negotiation protocol, decision model, coordination model
Web service QoS profile	Pre-determined, non-negotiable, not-customizable	Two types of QoS attributes in the profile: negotiable (flexible value), non-negotiable (pre-determined, fixed value)
Disciplinary influences	Well-developed body of knowledge available in optimization area	Automated negotiation approaches still in their infancy; very complex, and multi-disciplinary research area
Application of the original disciplines for compposite service selection	Optimization techniques have been specifically applied for composite service selection, considering the particular constraints and challenges involved in the WSC domain	Need for more inter-disciplinary approaches drawing on game theory, economics (auctions), and coordination theory.
Performed experiments	Typically include four to five QoS attributes in the scenarios.	Fewer QoS attributes included in the experiments. Generally not more than two, in order to manage complexity.
Facilitating tools	(M)ILP :available solvers, GA: well known algorithms, easy to implement	Automated negotiation support tools lacking. Domain-specific tools exist that assist human negotiators, e.g. Kasbah [38], Negoisst [51], CyberSettle[a]

[a] http://www.cybersettle.com/

are possible, a negotiation-based approach involving two or more agents engaging in an iterative communication and decision making process can become feasible. A summary of the comparison of optimization and negotiation-based approaches is presented in Table 13.1.

13.7 Conclusion

As the number of available web services with similar functionality but varying QoS attribute values increases, the problem of discovering and selecting the best web services for composing enterprise applications becomes more challenging. We have presented a comprehensive review of some of the important optimization- and negotiation-based approaches to the service selection problem in WSC. A brief overview of two of the hybrid models is also provided. In conclusion, optimization-based approaches assumption about a pre-determined profile is a hindrance in applying them for real service selection scenarios, considering the dynamic execution environment of web services. While in negotiation-based approaches this assumption is relaxed, the need for a complex decision model that guides the software agents in conducting fully automated negotiation makes their application unrealistic at least for the near future. There is a need for service selection techniques with more realistic assumptions in their service specification, discovery and selection models to make them relevant and useful for the emerging real world service selection scenarios.

References

1. Agarwal, V., Jalote, P.: From specification to adaptation: an integrated QoS-driven approach for dynamic adaptation of web service compositions. In: IEEE International Conference on Web Services (ICWS), pp. 275–282 (2010)
2. Alrifai, M., Skoutas, D., Risse, T.: Selecting skyline services for QoS-based web service composition (2010). doi:10.1145/1772690.1772693
3. Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Nakata, T., Pruyne, J., Rofrano, J., Tuecke, S., Xu, M., (2004¢ 2007), O.G.F.O.: Web services agreement specification (WS-Agreement). Technical report (2007)
4. Ardagna, D., Pernici, B.: Adaptive service composition in flexible processes. IEEE Trans. Softw. Eng. **33**(6), 369–384 (2007)
5. Baryannis, G., Danylevych, O., Karastoyanova, D., Kritikos, K., Leitner, P., Rosenberg, F., Wetzstein, B.: Service Composition (2010). www.s-cube-network.eu/results/books/Bookv0.4.pdf
6. Berbner, R., Spahn, M., Repp, N., Heckmann, O., Steinmetz, R.: Heuristics for QoS-aware web service composition. In: International Conference on Web Services (ICWS '06), pp. 72–82 (2006)
7. Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., Orchard, D.: Web Services Architecture (2004). http://www.w3.org/TR/ws-arch/
8. Brogi, A., Corfini, S.: Behaviour-aware discovery of Web service compositions. Int J Web Serv Res **4**(3), 1–25 (2007)
9. Brogi, A., Corfini, S., Popescu, R.: Semantics-based composition-oriented discovery of web services. ACM Trans. Internet Technol. **8**(4), 1–39 (2008). doi:10.1145/1391949.1391953
10. Canfora, G., Penta, M.D., Esposito, R., Villani, M.L.: An approach for QoS-aware service composition based on genetic algorithms (2005). doi:http://doi.acm.org/10.1145/1068009.1068189
11. Carlson, S.E.: A general method for handling constraints in genetic algorithms. In: Proceedings of the Joint Conference on Information Science, Citeseer, pp. 663–667 (1995)
12. Chhetri, M.B., Lin, J., Goh, S.K., Yan, J., Zhang, J.Y., Kowalczyk, R.: A coordinated architecture for the agent-based service level agreement negotiation of Web service composition. In: Australian Software Engineering Conference (ASWEC), p. 10 (2006)

13. Comuzzi, M., Francalanci, C., Giacomazzi, P.: Trade-Off Based Negotiation of Traffic Conditioning and Service Level Agreements in DiffServ Networks. In: Chiara, F., Paolo, G. (eds.) International Conference on Advanced Information Networking and Applications, vol. 1, pp. 189–194. Taipei, Taiwan (2005)
14. Comuzzi, M., Pernici, B.: An architecture for flexible Web service QoS negotiation. In: IEEE International Enterprise Computing Conference (EDOC), pp. 70–79 (2005)
15. Comuzzi, M., Pernici, B.: A framework for QoS-based web service contracting. ACM Trans. Web **3**(3), 1–52 (2009). doi:10.1145/1541822.1541825
16. Di Nitto, E., Di Penta, M., Gambi, A., Ripa, G., Villani, M.: Negotiation of Service Level Agreements: An Architecture and a Search-Based Approach. In: Krämer, B., Lin, K.J., Narasimhan P (eds.) Service-Oriented Computing ¢ ICSOC 2007, vol. 4749, pp. 295–306. Springer, Berlin (2007). doi:10.1007/978-3-540-74974-5_24
17. El Haddad, J., Manouvrier, M., Rukoz, M.: TQoS: transactional and QoS-aware selection algorithm for automatic web service composition. IEEE Trans Serv Comput **3**(1), 73–85 (2010)
18. Erl, T.: SOA: Principles of Service Design. Prentice Hall, USA (2008)
19. Faratin, P., Sierra, C., Jennings, N.R.: Negotiation decision functions for autonomous agents. Int. Journal of, Robot Auton Syst **24**(3–4), 159–182 (1998)
20. Faratin, P., Sierra, C., Jennings, N.R.: Using similarity criteria to make issue trade-offs in automated negotiations. Artif Intell **142**(2), 205–237 (2002). doi:10.1016/s0004-3702(02)00290-4
21. Fonseca, C.M., Fleming, P.J.: Multiobjective optimization and multiple constraint handling with evolutionary algorithms. I. A unified formulation. IEEE Trans Syst Man Cybernetics Part A: Systems and Humans **28**(1), 26–37 (1998)
22. Foundation for Intelligent Physical Agents: FIPA contract net interaction protocol (2000). http://www.fipa.org/specs/fipa00029/SC00029H.pdf
23. Georgakopoulos, D., Papazoglou, M.P.: Service-Oriented Computing. MIT Press, Cambridge (2009)
24. Gimpel, H., Ludwig, H., Dan, A., Kearney, B.: PANDA: Specifying Policies for Automated Negotiations of Service Contracts. In: Orlowska, M.E., Weerawarana, S., Papazoglou, M.P., Yang J. (eds.) Service-Oriented Computing—ICSOC 2003, vol. 2910, pp. 287–302. Springer, Berlin (2003). doi:10.1007/978-3-540-24593-3_20
25. Grimm, S.: Discovery, Identifying Relevant Services. In: Semantic Web Services: Concepts, Technologies, and Applications. Springer, New York (2007)
26. Hao, Y., Zhang, Y., Cao, J.: Web services discovery and rank: an information retrieval approach. Future Gener Comput Syst **26**(8), 1053–1062 (2010). doi:10.1016/j.future.2010.04.012
27. Hilton, A.B.C., Culver, T.B.: Constraint handling for genetic algorithms in optimal remediation design. J.Water Resour. Planning Manag **126**(3), 128–137 (2000)
28. Hudert, S., Ludwig, H., Wirtz, G.: Negotiating SLAs-An approach for a generic negotiation framework for WS-agreement. Journal Grid Comput **7**(2), 225–246 (2009). doi:10.1007/s10723-009-9118-3
29. Jaeger, M.C., Muehl, G.: QoS-based selection of services: the implementation of a genetic algorithm. In: Communication in Distributed Systems (KiVS), 2007 ITG-GI Conference pp. 1–12 (2007)
30. Jaeger, M.C., Rojec-Goldmann, G., Muhl, G.: QoS aggregation for Web service composition using workflow patterns. In: Eighth IEEE International Enterprise Distributed Object Computing Conference (EDOC), pp. 149–159 (2004)
31. Jaeger, M.C., Rojec-Goldmann, G., Muhl, G.: QoS aggregation in web service compositions. In: IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE '05), pp. 181–185 (2005)
32. Jennings, N.R., Faratin, P., Lomuscio, A.R., Parsons, S., Wooldridge, M.J., Sierra, C.: Automated negotiation: prospects, methods and challenges. Group Decis Negot **10**(2), 199–215 (2001)
33. Jiuxin, C., Yongsheng, L., Junzhou, L., Bo, M.: Efficient multi-QoS attributes negotiation for service composition in dynamically changeable environments. In: IEEE International Conference on Systems Man and Cybernetics (SMC), pp. 3118–3124 (2010)

34. Keeney, R.L., Raïffa, H.: Decisions with multiple objectives: preferences and value tradeoffs. Cambridge University Press, Cambridge (1993)
35. Kim, J.B., Segev, A., Patankar, A., Cho, M.G.: Web services and bpel4ws for dynamic ebusiness negotiation processes. In: Conference on Web Services, ICWS, vol. 3, pp. 111–117. Citeseer (2003)
36. Lecue, F., Mehandjiev, N.: Towards scalability of quality driven semantic web service composition. In: IEEE Int Conf Web Serv (ICWS), pp. 469–476 (2009)
37. Ma, Y., Zhang, C.: Quick convergence of genetic algorithm for QoS-driven web service selection. Comput Netw **52**(5), 1093–1104 (2008). doi:10.1016/j.comnet.2007.12.003
38. Maes, P., Guttman, R.H., Moukas, A.G.: Agents that buy and sell. Commun. ACM 42(3), 81-ff (1999). doi:10.1145/295685.295716
39. McIlraith, S., Son, T.C.: Adapting golog for composition of semantic web services. In: International Conference on the Principles of Knowledge Representation and Reasoning, pp. 482–496. Citeseer (2002)
40. Medjahed, B., Atif, Y.: Context-based matching for Web service composition. Distrib Parallel Databases **21**(1), 5–37 (2007). doi:10.1007/s10619-006-7003-7
41. Menasce, D.A., Casalicchio, E., Dubey, V.: On optimal service selection in service oriented architectures. Perform. Eval. **67**(8), 659–675 (2010). doi:10.1016/j.peva.2009.07.001
42. Michalewicz, Z.: A survey of constraint handling techniques in evolutionary computation methods. In: Proceedings of the 4th Annual Conference on Evolutionary Programming, pp. 135–155. The MIT Press, Cambridge (1995)
43. Mueller, H.J.: Negotiation principles. In: O'Hare, G.M.P., Jennings, N.R. (eds.) Foundations of Distributed Artificial Intelligence, pp. 211–229. Wiley, New York (1996)
44. Mukhija, A., Dingwall-Smith, A., Rosenblum, D.S.: QoS-Aware service composition in Dino. In: Fifth European Conference on Web Services, pp. 3–12 (2007)
45. O'Sullivan, J., Edmond, D., ter Hofstede, A.: What's in a service? Distrib Parallel Datab **12**(2), 117–133 (2002). doi:10.1023/a:1016547000822
46. Parra-Hernandez, R., Dimopoulos, N.J.: A new heuristic for solving the multichoice multidimensional Knapsack problem. IEEE Trans Syst Man Cybernetics A **35**(5), 708–717 (2005)
47. Rao, J., Su, X.: A Survey of Automated Web Service Composition Methods (2005). doi:10.1007/978-3-540-30581-1_5
48. Richter, J., Baruwal Chhetri, M., Kowalczyk, R., Bao Vo, Q.: Establishing composite SLAs through concurrent QoS negotiation with surplus redistribution. Concurrency Comput Prac Experience (2011). doi:10.1002/cpe.1727
49. Richter, J., Chhetri, M.B., Kowalczyk, R., Bao Quoc, V., Talib, M.A., Colman, A.: Utility decomposition and surplus redistribution in composite SLA negotiation. In: IEEE International Conference on Services Computing (SCC), pp. 627–630 (2010)
50. Rosenberg, F., Celikovic, P., Michlmayr, A., Leitner, P., Dustdar, S.: An End-to-End approach for QoS-Aware service composition. In: IEEE International Enterprise Distributed Object Computing Conference (EDOC '09), pp. 151–160 (2009)
51. Schoop, M., Jertila, A., List, T.: Negoisst: a negotiation support system for electronic business-to-business negotiations in e-commerce. Data Knowl Eng **47**(3), 371–401 (2003). doi:10.1016/s0169-023x(03)00065-x
52. UDDI Consortium: UDDI executive white paper (2001). www.uddi.org/pubs/UDDI_Executive_White_Paper.pdf.
53. Ul Haq, I., Paschke, A., Schikuta, E., Boley, H.: Rule-based validation of SLA choreographies. J Supercomput, pp. 1–22 (2010). doi:10.1007/s11227-010-0492-1
54. Wang, P.: QoS-aware web services selection with intuitionistic fuzzy set under consumer¢¢s vague perception. Expert Syst Appl **36**(3, Part 1), 4460–4466 (2009). doi:10.1016/j.eswa.2008.05.007
55. Wiesemann, W., Hochreiter, R., Kuhn, D.: A stochastic programming approach for QoS-aware service composition. In: IEEE International Symposium on Cluster Computing and the Grid (CCGRID '08), pp. 226–233 (2008)

56. Wilkes, J.: Utility Functions, Prices, and Negotiation. In: Buyya, R., Bubendorfer, K. (eds.) Market-Oriented Grid and Utility Computing. Wiley, Hoboken (2009)
57. Yan, J., Kowalczyk, R., Lin, J., Chhetri, M.B., Goh, S.K., Zhang, J.: Autonomous service level agreement negotiation for service composition provision. Future Gener Comput Syst **23**(6), 748–759 (2007). doi:10.1016/j.future.2007.02.004
58. Yan, J., Zhang, J., Lin, J., Chhetri, M.B., Goh, S.K., Kowalczyk, R.: Towards autonomous service level agreement negotiation for adaptive service composition. In: 10th International Conference on Computer Supported Cooperative Work in Design (CSCWD '06), pp. 1–6 (2006)
59. Yu, Q., Bouguettaya, A.: Multi-attribute optimization in service selection. World Wide Web **15**(1), 1–31 (2012). doi:10.1007/s11280-011-0121-9
60. Yu, T., Zhang, Y., Lin, K.J.: Efficient algorithms for web services selection with end-to-end QoS constraints. ACM Trans. Web (TWEB) **1**(1), 6 (2007)
61. Zadeh, L.A.: Similarity relations and fuzzy orderings. Inf Sci **3**(2), 177–200 (1971). doi:10.1016/s0020-0255(71)80005-1
62. Zeng, L., Benatallah, B., Ngu, A.H.H., Dumas, M., Kalagnanam, J., Chang, H.: QoS-Aware middleware for web services composition. IEEE Trans. Softw. Eng. **30**(5), 311–327 (2004). doi:10.1109/tse.2004.11
63. Zulkernine, F., Martin, P., Craddock, C., Wilson, K.: A policy-based middleware for web services SLA negotiation. In: IEEE International Conference on Web Services ICWS, pp. 1043–1050 (2009)
64. Zulkernine, F.H., Martin, P.: An adaptive and intelligent SLA negotiation system for web services. IEEE Trans. Serv. Comput. 4(1), 31–43 (2011)