A Market-Based Coordinated Negotiation for QoS-Aware Service Selection

Claudia Di Napoli[1], Dario Di Nocera[2], Paolo Pisa[3], and Silvia Rossi[3](✉)

[1] Istituto di Calcolo e Reti ad Alte Prestazioni - C.N.R., Naples, Italy
claudia.dinapoli@cnr.it

[2] Dipartimento di Matematica e Applicazioni, University of Naples "Federico II", Napoli, Italy
dario.dinocera@unina.it

[3] Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione, University of Naples "Federico II", Napoli, Italy
silvia.rossi@unina.it

Abstract. The provision of Service-Based Applications (SBAs) will be driven by market-oriented mechanisms, and the market value of an application will depend not only on its functionality, but also on the value of "Quality of Service" (QoS) parameters affecting its performance. These parameters are not static properties since they may vary according to the provision strategies of providers as well as the demand of users having their own preferences on the application QoS values. In this paper we propose a market-based negotiation mechanism among service providers and a user requesting a QoS-aware SBA. It allows to take into account the variability of service QoS attribute values typical of the future market of services, as well as to dynamically set the length of the negotiation process that is usually very time-consuming especially in the context of SBAs.

Keywords: Service-oriented architectures · Artificial economies/markets · Negotiation · Quality of service · Service selection

1 Introduction

It is well recognized that Service Based Applications (SBAs) will be provided with Quality of Service (QoS) attributes that take account of service not functional properties (NFPs) such as price, response time, reliability, reputation, and so on [1]. QoS-aware SBAs are composed of autonomous and independent services that are provided with different quality attributes representing their NFPs, and they are required by users that have their own preferences over the values of these attributes. Hence, in order to deliver QoS-aware SBAs, the attribute values of their component services have to meet the user requirements, once aggregated.

DOI: 10.1007/978-3-319-13218-1_3

Nevertheless, different users may have different QoS requirements for the same application, as well as QoS attribute values for the same service may change in time according to dynamic circumstances affecting service provision strategies.

In this context, it becomes crucial to provide service-oriented infrastructures with mechanisms enabling the selection of services with suitable QoS attribute values so that QoS requirements can be satisfied when forming new value-added applications through service composition. Such mechanisms should allow to manage the dynamic nature of both provided QoS values, and user's QoS requirements.

In this paper we propose a negotiation-based mechanism among service providers and a service consumer to select the suitable services to compose QoS-aware SBAs through a market-based provision mechanism. The negotiation-based selection mechanism allows for the selection of services according to the values of their quality attributes so that, once aggregated, they meet the user quality constraints/preferences. The use of a negotiation-based mechanism allows to take into account the variability of service QoS attribute values typical of the future market of services since service providers may change these values during the negotiation according to their own provision strategies.

Since negotiation can be computationally expensive, a set of experiments was carried out to assess the impact of such coordinated negotiation mechanism on the success rate of the composition process, and to collect useful information to drive service consumers decisions about whether to proceed with the negotiation under specific conditions, or not.

The paper is so organized: Sect. 2 introduces the problem of QoS-aware service composition and provides some related works. Section 3 describes the proposed coordinated negotiation mechanism, together with the adopted strategies for the negotiators. Section 4 presents the case study and discusses the collected experimental results. Conclusions and future work are reported in Sect. 5.

2 QoS-Aware Service Composition

In a market of services, users will issue a request for an SBA specifying the functionality of each service component, their functional dependence constraints, and the value(s) of the quality attribute(s) they want the application to provide. The request is described by a directed acyclic graph, called an *Abstract Workflow* (AW), and by a quality attribute value representing the required *QoS* for the application. AW nodes represent the required functionalities, called *Abstract Services* (ASs), and AW arcs represent control and data dependencies among nodes.

It is assumed that for each AS a set of *Concrete Services* (CS) will be available on the market, each one provided by a specific *Service Provider* (SP) with QoS attributes whose values are set by the corresponding SP dynamically.

The user request is managed by a software entity, named *Service Compositor* (SC), responsible for the selection of CSs whose attribute values, once aggregated, satisfy the QoS required by the user. The selection is modeled as a negotiation process over the service quality attributes among the SC and the SPs available to provide their services. SPs issue their offer to the SC by specifying a reference to the CS together with the value of the QoS attribute they can provide the service with at that time. If the negotiation is successful, then the user request can be satisfied by instantiating the AW with the CSs having the suitable QoS value. The *Instantiated Workflow* (IW) represents the requested application ready to be executed.

2.1 Related Works

Several efforts have been carried out in the areas of QoS-based service selection for Service Based Applications [2].

Some works propose algorithms to select service implementations relying on the optimization of a weighted sum of global QoS parameters as in [3] by using Integer Linear Programming (ILP) methods. Nevertheless, ILP-based algorithms for selecting services are suitable when QoS data are accurate and the problem size is small (i.e. with a limited number of nodes for the Abstract Workflow and a limited number of potentially available services for each node) due to the ILP high complexity [4]. In such cases, instead of optimal solution procedures, heuristic algorithms are have been proposed in the literature [5]. In this context also Genetic Algorithms have been proposed to address scalability problems as in [6,7]. Approximation-based approaches are more efficient than linear approaches as they can handle large number of services better than linear methods. However, they suffer from lack of ability to find optimal solutions since they can be discarded during the elimination process using these approaches.

In [8] local constraints are included in the linear programming model used to satisfy global QoS constraints. In [9] Mixed Integer Programming is used to find the optimal decomposition of global QoS constraints into local constraints representing the service skyline for each service class, so allowing to prune the service candidates that are not likely to be part of the optimal solution.

Typically, these works rely on static approaches assuming that QoS parameters of each service do not change during the selection process, i.e. they are predetermined, and focus on optimality and performances of the provisioning methods. Such approaches do not take into account the possibility to dynamically change the provided QoS values during the selection process that represents the basic motivation for the approach proposed in our work. Other approaches rely on negotiation mechanisms to select services according the QoS values [10,11]. In most of these approaches negotiation occurs when the service provider has already been selected, and it negotiates the values of the service parameters values it provides for the service. So, the negotiation process is one-to-one between service requester and the selected service provider [3].

Other negotiation-based approaches use negotiation as a mechanism to dynamically select the appropriate the service provider whose provided services best matches the service requester's non-functional requirements [12]. But usually negotiation is carried out for each required service independently from the others. So negotiation consists of multiple negotiation sub-processes each one associated with one once of the required composite service. Each negotiation sub-process, in turn, may include multiple negotiation threads, one for each candidate provider, to choose the best service for the specific component service.

The work presented, in this paper, proposes a coordinated negotiation mechanism, where negotiations occurs concurrently with all providers of the different required services in the composition. Coordination occurs at each negotiation step when the aggregated QoS values offered by different SPs are collectively evaluated to decide whether to accept or not a set of offers, so to take into account the dependencies among different negotiation processes due to the fact that in a composition of services the attributes values for one services cannot be determined independently from the other services in the composition.

3 The Coordinated Negotiation Mechanism

The negotiation process between two agents x and y is a bilateral interaction that consists of an alternate succession of offers and counteroffers. The process continues either until an offer is accepted by the other agent, or one of the agents terminates the interaction (e.g., because of a deadline). An agent x accepts an offer j of y if the value of the utility the agent x obtains for that offer is greater than the utility value of the counteroffer the agent x would send back, i.e. $U_x(j_y(t)) \geq U_x(j_x(t+1))$ [13].

In order to prepare a counteroffer, an agent uses a set of tactics to generate new values for each negotiated object [14]. Of course, both agents must be provided with strategies to formulate offers and counteroffers, and they must be equipped with algorithms to evaluate the received offers.

In this paper we consider a modified negotiation mechanism, based on an iterative protocol, allowing only the SPs to formulate new offers, and the SC only to evaluate them. The rationale of this choice is twofold: on one hand it makes it possible to simulate what happens in a real market of services where an SC does not have enough information on the SPs strategies to formulate counteroffers; on the other hand it takes into account that the offers for a single functionality cannot be evaluated independently from the ones received for the other functionalities. In other words, negotiating over the attributes of the single AS cannot be done independently from each other. In fact, the proposed negotiation mechanism allows both to negotiate with the SPs providing services for the same required functionality in the AW, and, at the same time, to evaluate if the aggregated QoS value of the received offers meets the QoS requirement specified in the user request, to decide whether or not to accept the offers.

3.1 The Negotiation Protocol

In order to meet the user's requirements, an iterative negotiation protocol, based on the Contract Net Iterated Protocol [15], is adopted. The negotiation occurs between the SC, that is the *initiator* of the negotiation, and the SPs available for each AS of the AW, and it may be iterated for a variable number of times until a *deadline* is reached or the negotiation is successful (see Fig. 1). Each iteration

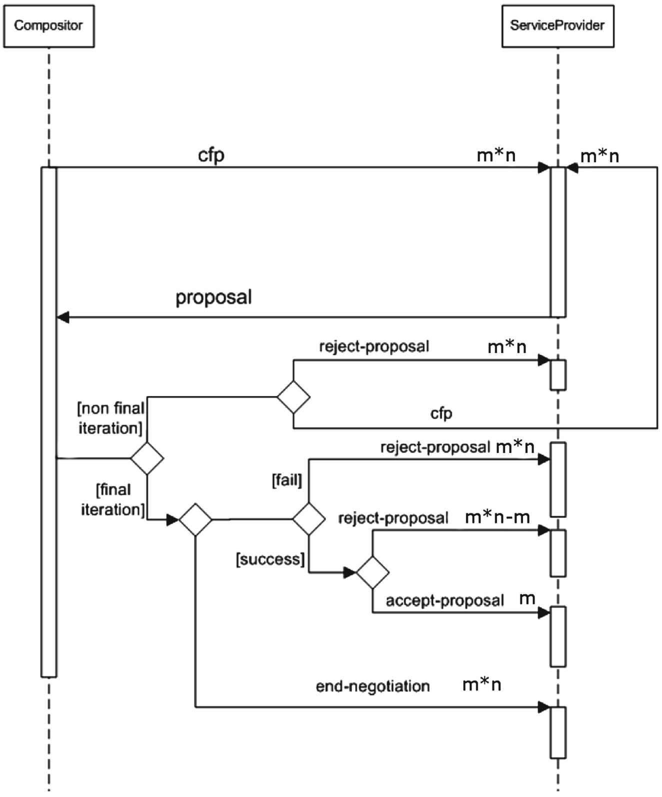


Fig. 1. The negotiation protocol.

is referred to as a negotiation *round*, and the deadline is the number of allowed rounds. In this protocol the SC may set the deadline according to estimates of parameters influencing the negotiation progress (see Sect. 4), so the negotiation takes place for a variable number of iterations based on the specific situation occurring when a request is issued.

As shown in the Fig. 1, the SC prepares m call for proposals (*cfp*s), one for each AS in the AW to send to the set of n SPs available to take part in the negotiation for that AS. So, the total number of *cfp*s sent at each round is $m * n$.

After waiting for the time set to receive offers (*expiration time* of a negotiation round), the SC checks if there are offers for each AS; if not, it declares a failure since it is not possible to find a CS corresponding to each AS. Otherwise, it evaluates the received offers, and, according to the result of the evaluation (see Sect. 3.2), it performs one of the following actions:

– if the aggregated QoS value of the received offers does not meet the user's QoS requirements, it asks for new offers by sending again $m * n$ *cfp*s, so starting another negotiation round;

– if the aggregated QoS value of the received offers meets the user's QoS requirements, it selects the best set of offers, in terms of its own utility, i.e. it accepts the offers sent by the corresponding SPs (one for each AS), so ending the negotiation successfully.
– if the deadline is reached without a success, the SC declares a failure to all SPs that took part in the negotiation.

3.2 Service Compositor

The SC receives service offers at each negotiation round and, once checked that there is at least one offer for each AS, it evaluates if the global QoS constraints specified by the user are met. The constraints are intended to be upper bounds for the aggregated values obtained by the offered QoSs, and the evaluation function is a solver of a Integer Linear Programming problem. We decided to use this global optimization approach, and not to investigate sub-optimal approaches since we are mainly interested in the evaluation of the impact of the coordinated negotiation approach on the probability to succeed. The ILP problem is formulated as follows. There are $n * m$ decision variables $x_{i,j}$ where i identifies one of the m ASs and the j identifies one of the n SPs compatible with the i-th AS. Such variables assume value 1 if the j-th SP is selected for the AS i, 0 otherwise. Exactly one SP has to be selected for each AS, and so the sum of $x_{i,j}$ for a specified AS i, must be equal to 1. This constraint holds for all ASs, so:

$$\sum_{j=1}^{n} x_{i,j} = 1, \forall i = 1, \ldots, m \tag{1}$$

Assuming a multidimensional QoS $(Q_1, \ldots, Q_r)$, the r-tuple $(q^1_{i,j}, \ldots, q^r_{i,j})$ of offered values is associated to each corresponding SP identified by $x_{i,j}$.

To check whether each QoS constraint has been satisfied, the values of the parameters $q^k_{i,j}$ offered by each selected SP, once aggregated, must not exceed the user upper bound Q_k:

$$aggrFun_i(\sum_{j=1}^{n} x_{i,j} * q^k_{i,j}) \leq Q_k, \forall k = 1, \ldots, r \tag{2}$$

The aggregation function $aggrFun$ for the QoS parameters depends on the type of the parameter. Typically, additive (e.g., price and execution time) and multiplicative (e.g., reliability and availability) parameters are considered [3], so $aggrFun$ is either a sum or a multiplication over the m ASs.

Once solutions that satisfy the constraints of Eq. 2 are found, the SC evaluates the overall utility [9]:

$$U(SC) = \sum_{k=1}^{r} \frac{Q_{max'(k)} - aggrFun_i(\sum_{j=1}^{n} x_{i,j} * q^k_{i,j})}{Q_{max'(k)} - Q_{min'(k)}} w_k \tag{3}$$

where, w_r is a weight for the specific QoS, $Q_{max(i,k)} = max(q^k_{i,j})$, $Q_{max'(k)} = aggrFun_k(Q_{max(i,k)})$, $Q_{min'(k)} = aggrFun_k(Q_{min(i,k)})$; i.e., $Q_{max'(k)}$ aggregates the maxima of the offers received for each AS, and $Q_{min'(k)}$ the corresponding local minima. The objective function is a maximization of Eq. 3. Hence, the SC selects the combinations of offers with the maximum utility among the ones that satisfy the constraints.

3.3 Service Provider

According to [14], time dependent and resource dependent strategies are two important classes of negotiation tactics in service-oriented domains, already used for modeling B2B interactions [8]. Time dependent strategies model the interactions of agents with deadlines for making deals. Usually, as the deadline approaches the agent is more willing to concede in utility. The tactics are implemented as time dependent functions, typically exponential or polynomial functions, classified as *boulware* or *conceder* tactics [14]. In the first case, the agent proposes values near to the initial offer until the deadline approaches, then it will propose its *reservation value* that represents the offer with the minimum utility the agents is able to provide within its negotiation set. In the second case, the agent will approach its reservation value sooner than the previous case.

Resource dependent strategies are similar to the time dependent ones, but the domains of functions modelling the tactics are the available resources, so it is necessary to evaluate the available resources w.r.t. the received requests to generate new offers.

In a previous work [16], a provider agent concession strategy was modeled as a monodimensional Gaussian function where the dimension represents a single negotiated QoS attribute. The use of Gaussian distributions allows to simulate the stochastic behaviour of service providers with zero-intelligence that can be used to approximate the trends of a volatile and open market of services [17]. In the present work, the same strategies are used, that are both time and resource dependent, and take into account the *computational load* of the SP, and the *computational cost* of the provided service. The computational load accounts for the provider workload in terms of the amount of service implementations it will deliver; while the computational cost represents a measure of the service complexity, so that the more complex the service is the higher its expected cost is.

The negotiation strategy is modeled, for each SP, by a Gaussian distribution that represents the probability distribution of the offers in terms of the provider's utility. As shown in Fig. 2, the mean value of the Gaussian $maxU$ represents the best offer the SP may propose in terms of its own utility having the highest probability to be selected; the standard deviation σ represents the attitude of the SP to concede during negotiation, and it is given by $\sigma_{i,j} = maxU_{i,j} - maxU_{i,j} * percent_{i,j}$, where $percent \in [0, 1]$ represents the concession percentage of the SP with respect to its own utility.

The negotiation set for the SP is $[maxU - \sigma; maxU]$, where $maxU - \sigma$ is the reservation value. The parameter σ varies from SP to SP providing the same AS,

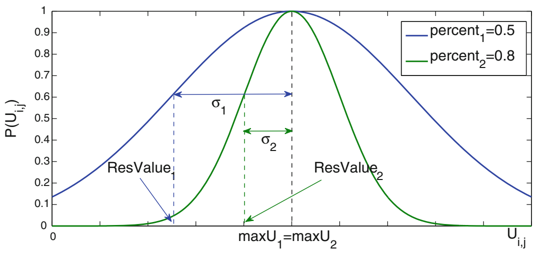


Fig. 2. An example of probability functions to compute new offers.

so that the lower its computational load (in terms of available resources) is, the more it is available to concede in utility, and the lower its reservation value is.

In Fig. 2 the functions associated to two different SPs for the same AS are reported. The best offer is the same for both SPs (i.e. $maxU_1 = maxU_2$) since it is assumed that services providing the same functionality have the same utility value for all the providers of that service, while their concession strategies are different according to their workload when the negotiation takes place. In the reported case σ_1 is greater than σ_2 meaning that SP_1 has a lower computational load than SP_2, so it concedes more in utility than SP_2.

At each negotiation round, each SP generates, following its distribution, a new utility value corresponding to a new offer. If this value is lower than the one offered in the previous round and within the negotiation set, then the SP proposes the new value. Note that values generated in the set $[maxU; maxU + \sigma]$ will be specularly mapped to the corresponding values within the negotiation set $[maxU - \sigma; maxU]$ with the same probability to be selected. If the new generated utility value is higher than the one offered at the previous round, or it is outside the negotiation set, the SP proposes the same value offered in the previous round.

This strategy allows to simulate different and plausible behaviors of providers that prefers not having a consistent loss in utility, even though by increasing the number of negotiation rounds the probability for the SP to move towards its reservation value increases.

4 A Case Study

A set of experiments was carried out in order to determine weather the coordinated mechanism affects the negotiation probability of success/failure, and to evaluate the impact of the number of SPs and ASs on the negotiation progress. The experiments were designed to extract information that can be used by the SC to understand negotiation trends according to the current market situation.

In the experiments, we considered a single QoS attribute, that is the price. So, the QoS aggregation function is additive in the number of ASs in the AW, and it does not depend on the structure of the AW, i.e., on the functional precedence relations among ASs. For this reason the nature of the arcs in the AW (representing sequential, parallel constructs, and so on) is not taken into consideration in the experiments.

4.1 The Price Parameter

Considering the price the only parameter to negotiate, the utility value for the SP is just the price of the service it offers. This means that the $maxU$ value is the $bestPrice$ in terms of the SP utility, and an SP offer is $Price_{i,j}$. The lowest price that the SP can offer is $bestPrice - \sigma$, that represents its reservation price.

With this setting, assuming there are m ASs in the required AW, and n SPs for each AS, the linear programming problem is formulated as in Sect. 3.2, where Eqs. 2 and 3 are instantiated follows:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} x_{i,j} * Price_{i,j} \leq reqPrice \tag{4}$$

$$U(SC) = \frac{Q_{max'(Price)} - \sum_{i=1}^{m}\sum_{j=1}^{n} x_{i,j} * Price_{i,j}}{Q_{max'(Price)} - Q_{min'(Price)}} \tag{5}$$

where, $reqPrice$ is the maximum price the user specified as hard constraint, expressed by Eq. 4.

The SC utility for each received offer j is given by Eq. 5. If there is a combination of offers that satisfies the constraint, the linear programming solver identifies it. It is assumed that the more complex the functionality a service provides, the higher its "market price" is. This price is also the one with the maximum utility for all SPs providing that functionality, i.e., it represents the $bestPrice$ for all the SPs. It is reasonable to assume that the variability in prices for different ASs is proportional to the complexity of the provided functionality. To simulate the variability of prices for services providing different functionality in terms of their complexity, a parameter k is used. The more complex the functionality provided by a service is the higher the value k is.

The k parameter is set to be equal for all providers of the same service, meaning that services providing the same functionality have the same market price. In fact, the value k determines the mean value of the Gaussian distribution, i.e., the price that most likely will be proposed for the service that is given by:

$$bestPrice_i = \frac{reqPrice * k_i}{m} \qquad i \in [1, \ldots, m] \tag{6}$$

where, m is the number of ASs in the AW, $k \leq 1$ for SPs providing less complex services, and $k > 1$ for SPs providing more complex services.

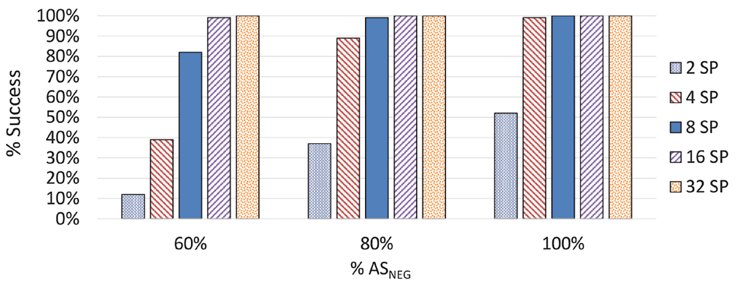
If for each AS $k \leq 1$, then the first offers will be at most equal to $reqPrice/m$ (i.e., the price is equally distributed among ASs). In this case, the QoS constraint is fulfilled at the first negotiation round. If for all the ASs $k > 1$, then at the first round there is not any combinations of offers leading to the constraint satisfaction, i.e., no *feasible* solution exists. The lack of a feasible solution implies that the SC has no chance of getting an instantiation of the workflow, so it has to iterate the negotiation.

The value of $bestPrice_i$ in Eq. 6 takes into account both the computational cost of the offered service, and that the requested price $reqPrice$ is not "unreasonable" compared to the market price of the required ASs in the AW.

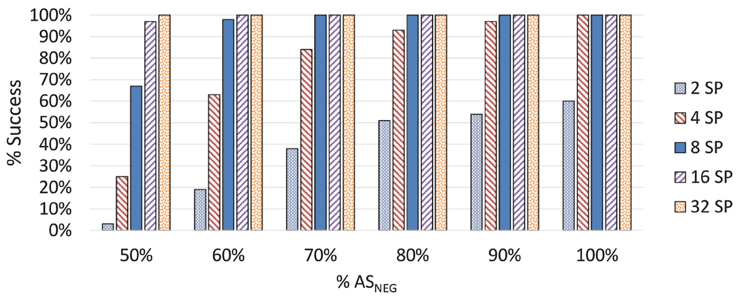
4.2 Numerical Evaluation of the Negotiation Trends

In order to model reasonable market situations, we fixed the k_i values for each AS so that the average value is equal to 1.5. In fact, this setting models a market configuration where it is not possible to obtain a success at the first negotiation round, but there are still good chances to reach a success during the process.

We considered two AW configurations including respectively 5 and 10 ASs. The ASs have a different a default price, $bestPrice_i$, determined by the following values of k_i: 2.4, 2.0, 1.3, 1.0 and 0.8 for the case of 5 ASs, and 2.6, 2.4, 2.2, 2, 1.7, 1,3, 1, 0.8, 0.6 and 0.4 for the case of 10 ASs. For each AS, the corresponding SPs will send as initial offer a price in the neighborhood of $bestPrice_i$ $[bestPrice_i - 5\,\%, bestPrice_i]$. The concession percentage value of each SP, $percent_{i,j}$, randomly varies in the range [0.5, 1.0], so including SPs



(a) AW with 5 ASs.



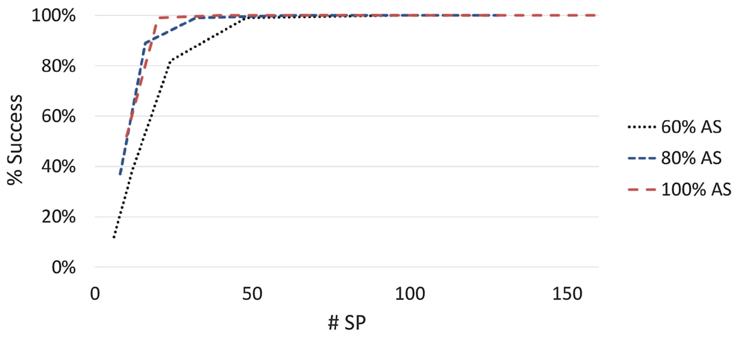
(b) AW with 10 ASs.

Fig. 3. Percentage of successes varying the percentage of ASs in the AW with negotiable QoS, and the number of SPs for each AS.

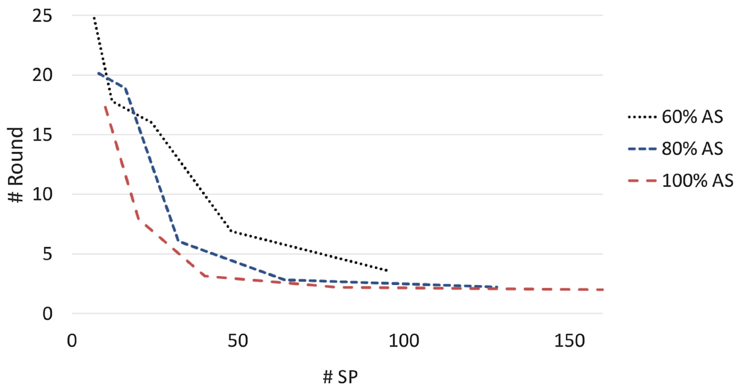
with the maximum computational load that are not willing to concede (i.e., $percent_{i,j} = 1$), and SPs with a low computational load willing to concede until a reservation price that is half of their *bestPrice*. The maximum number of negotiation rounds is 100, and the result of each experiment is mediated on 100 runs.

In a first set of experiments we evaluated the percentage of obtained successful negotiations in the case the negotiation occurs only for a subset of ASs (referred to as AS_{NEG}), and varying the number of SPs for each AS. This configuration models a market situation including service types whose QoS values are not negotiable.

As expected, the percentage of successes increases by increasing both the percentage of AS_{NEG}, and the number of SPs for each AS. This holds both for the case of 5 ASs (see Fig. 3a) and for the case of 10 ASs (see Fig. 3b). In particular, the fewer AS_{NEG} are in the AW, the higher the percentage of failures in the negotiation is, meaning that the probability of successful negotiations



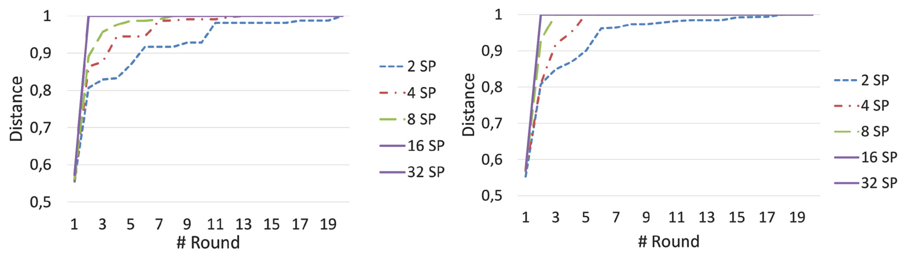
(a) Percentage of successes.



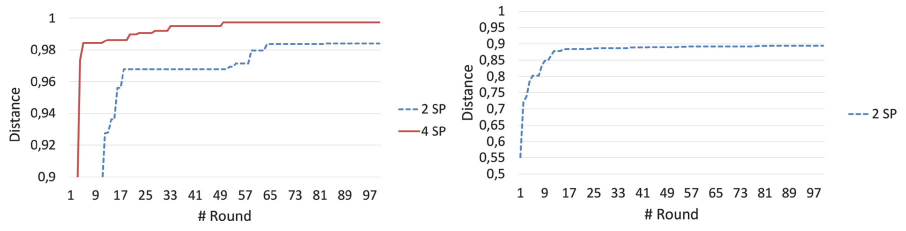
(b) Length of successful negotiations.

Fig. 4. % of successes and # of rounds w.r.t. the number of negotiating SPs, varying the percentage of AS_{NEG} in the case of 5 AS decreases, so, if the QoS of all the types of services is negotiable such probability increases. At the same time, the probability of successful negotiations increases by increasing the number of SPs for each AS. This is due to the fact that increasing the number of SPs compensates the number of SPs that do not negotiate at all. Furthermore, in our settings, by increasing the number of ASs in the AW, the number of SPs involved in the negotiation increases. This means that, with respect to the same percentage of AS_{NEG} and the same number of SPs for each AS (see Figs. 3a and b), the percentage of success increases by increasing the number of ASs in the AW. However, this does not directly means that the success rate increases by increasing only the number of SPs because of the increasing complexity of the AW (i.e., the number of ASs). For example, in Fig. 3a in the configuration with 4 SPs for 5 ASs all negotiating (20 SPs in total) we have a success rate of 99 %, while in Fig. 3b in the configuration with 2 SPs for 10 ASs all negotiating (20 SPs in total) we have a success rate of 60 %. This is also true when not all the SPs are involved in the negotiation.

In Fig. 4a we plotted the percentage of success with respect to the total number of SPs involved in the negotiation process, varying the percentage of AS_{NEG} for the case of 5 ASs (the case of 10 ASs has similar trends). Considering a fixed number of SPs, it can be noticed that the more the negotiating SPs are spread among ASs, the higher is the probability of success. Moreover, we evaluated length of successful negotiation (i.e., the number of rounds necessary to reach a success). As shown in Fig. 4b, the length decreases by increasing the number of SPs, and for the same number of negotiating SPs (e.g., 50) the length



(a) Distance in a case of success for 5 and 10 ASs.



(b) Distance in a case of failure for 5 and 10 ASs.

Fig. 5. Normalized distance of the best service aggregation from the user's constraint.

of negotiation decreases increasing the number of AS_{NEG}. This confirms that it is worth negotiating, when composition of services are required, and that it is worth negotiating with all available SPs in order to increase the probability of successful negotiation.

Finally, in Fig. 5 we plotted the distance of the best combination of offers from the user's QoS constraint, at each negotiation round, normalized in the range [0,1], to analyze the negotiation trends in time. In particular, Fig. 5a reports such distance in five cases of success (respectively for 5 ASs and 10 ASs in the AW) showing that the greater the number of negotiating SPs is, the faster the success is reached. Figure 5b show the same distance in cases of failures obtained respectively with 2 and 4 SPs for 5 ASs, and with 2 SPs for 10 ASs, showing that when a plateau is reached it is very likely the negotiation ends with failure.

5 Conclusions

In the present work the use of software agent negotiation is proposed as a means to select service implementations required by a SBA by taking into account the Quality of Service that providers offer for their services, and the end-to-end QoS requirements expressed by a user requesting the application. The use of negotiation allows to address the limitations of several approaches for the QoS-based selection of services in composition of services that are based on the assumption that QoS provided for the required services are static during the selection process. This assumption is not realistic in service provision scenarios of the future that are likely to be regulated by market-based mechanisms. It is necessary to allow service providers to change dynamically their provision strategies, so changing the value of QoS parameters according to market trends while the selection takes place.

The proposed iterative negotiation mechanism allows providers to change their offers at each iteration so that, in principle, they could change their provision strategies to be more competitive in the market. The experiments carried out showed that it is worth to negotiate with all available SPs in order to increase the probability of successful negotiation. Of course, the adopted CNP-based iterative negotiation protocol have a lot of communication overhead due to the broadcast of the *cfp*s to all the available SPs, so its performances may degrade drastically when the number of SPs and the number of ASs increases. Some works in the literature [18] propose learning-based mechanisms to help selecting the most promising agents so limiting the number of negotiating agents. But, in a market of services it is not possible to assume that a promising provider will keep on sending promising offers, because a less promising provider may change its strategy in the meantime. In our approach, the increase in communication costs is partially compensated by the fact that, as shown in the experiments, by increasing the number of SPs the success rate of the negotiation increases. So, the overhead due to the communication cost is partially compensated by a decrease in the negotiation length, i.e. its overall computational cost. The analysis of the negotiation progress, in different configurations, allows to evaluate the possibility to stop negotiation if the distance of the aggregated best offers from the user's QoS constraint is not improving. This is an useful feature when adopting computationally expensive mechanisms like negotiation in service-based application settings.

Furthermore, the coordinated negotiation mechanism allows to evaluate aggregated offers through a linear programming solver, so tackling the problem that when selecting services whose aggregated QoS values have to meet end-to-end QoS requirements, the selection of one service cannot be made independently from the other services. This is even more crucial in case of multidimensional QoSs.

In order to reduce the communication overhead of the proposed mechanism, more experiments are planned to determine the "critical mass" of SPs that is worth to negotiate with in order to increase the probability of success in different market of configurations.

Acknowledgments. The research leading to these results has received funding from the EU FP7-ICT-2012-8 under the MIDAS Project (Model and Inference Driven - Automated testing of Services architectures), Grant Agreement no. 318786, and the Italian Ministry of University and Research and EU under the PON OR.C.HE.S.T.R.A. project (ORganization of Cultural HEritage for Smart Tourism and Real-time Accessibility).

References

1. Strunk, A.: Qos-aware service composition: a survey. In: Brogi, A., Pautasso, C., Papadopoulos, G.A. (eds.) 2010 IEEE 8th European Conference on Web Services (ECOWS), pp. 67–74. IEEE Computer Society (2010)
2. Zeng, L., Benatallah, B., Dumas, M., Kalagnanam, J., Sheng, Q.Z.: Quality driven web services composition. In: Proceedings of the 12th International Conference on World Wide Web, WWW '03, pp. 411–421. ACM, New York (2003)
3. Zeng, L., Benatallah, B., Ngu, A.H., Dumas, M., Kalagnanam, J., Chang, H.: Qos-aware middleware for web services composition. IEEE Trans. Softw. Eng. **30**(5), 311–327 (2004)
4. Yu, T., Zhang, Y., Lin, K.J.: Efficient algorithms for web services selection with end-to-end qos constraints. ACM Trans. Web **1**(1), 1–26 (2007)
5. Berbner, R., Spahn, M., Repp, N., Heckmann, O., Steinmetz, R.: Heuristics for qos-aware web service composition. In: Proceedings of the IEEE International Conference on Web Services, ICWS '06, pp. 72–82. IEEE Computer Society (2006)
6. Canfora, G., Di Penta, M., Esposito, R., Villani, M.L.: An approach for qos-aware service composition based on genetic algorithms. In: Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation, pp. 1069–1075. ACM (2005)
7. Liu, H., Zhong, F., Ouyang, B., Wu, J.: An approach for qos-aware web service composition based on improved genetic algorithm. In: 2010 International Conference on Web Information Systems and Mining (WISM), pp. 123–128 (2010)
8. Ardagna, D., Pernici, B.: Adaptive service composition in flexible processes. IEEE Trans. Softw. Eng. **33**(6), 369–384 (2007)
9. Alrifai, M., Risse, T.: Combining global optimization with local selection for efficient qos-aware service composition. In: Proceedings of the 18th International Conference on World Wide Web, WWW '09, pp. 881–890. ACM, New York (2009)

10. Paurobally, S., Tamma, S., Wooldridge, M.: A framework for web service negotiation. ACM Trans. Auton. Adapt. Syst. **2**(4), 14 (2007)
11. Siala, F., Ghedira, K.: A multi-agent selection of web service providers driven by composite qos. In: Proceedings of 2011 IEEE Symposium on Computers and Communications (ISCC), pp. 55–60. IEEE (2011)
12. 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.) ICSOC 2003. LNCS, vol. 2910, pp. 287–302. Springer, Heidelberg (2003)
13. Jennings, N.R., Faratin, P., Lomuscio, A.R., Parsons, S., Sierra, C., Wooldridge, M.: Automated negotiation: prospects, methods and challenges. Int. J. Group Decis. Negot. **10**(2), 199–215 (2001)
14. Faratin, P., Sierra, C., Jennings, N.R.: Negotiation decision functions for autonomous agents. Robot. Auton. Syst. **24**, 3–4 (1998)
15. Smith, R.G.: The contract net protocol: high-level communication and control in a distributed problem solver. IEEE Trans. Comput. **29**(12), 1104–1113 (1980)
16. Di Napoli, C., Pisa, P., Rossi, S.: Towards a dynamic negotiation mechanism for QoS-aware service markets. In: Pérez, J.B., et al. (eds.) Trends in Practical Applications of Agents and Multiagent. AISC, vol. 221, pp. 9–16. Springer, Heidelberg (2013)
17. Gode, D.K., Sunder, S.: Allocative efficiency of markets with zero-intelligence traders: market as a partial substitute for individual rationality. J. Polit. Econ. **101**(1), 119–137 (1993)
18. Deshpande, U., Gupta, A., Basu, A.: Performance enhancement of a contract net protocol based system through instance-based learning. IEEE Trans. Syst. Man Cybern. **35**(2), 345–357 (2005)