QoS-aware Service Selection Based on Genetic Algorithm

Hadjila Fethallah, Mohammed Amine Chikh, Dali Yahia Mohammed
Computer sciences department
UABT University – Tlemcen
Tlemcen Algeria
{f_hadjila, mea_chikh }@mail.univ-tlemcen.dz
mohammed.dali.y@gmail.com

Abstract— **The web service selection is a primordial step in the support of dynamic compositions, in fact, the presence of a set of services that provide the same functionality (inputs/outputs), but differ in QOS criteria, obliges us to adopt an optimization strategy in order to select the best ones. Several approaches of various natures (mono-objective, multi-objective...) were proposed to solve this problem. In this paper we propose a genetic algorithm which handles a single objective function and a set of global constraints that must be fulfilled. The obtained results are encouraging and merit to be continued.**

Keywords- web services; combinatory optimization; genetic algorithms;qualiy of service;service oriented architecture.

I. INTRODUCTION

The web service technology is an implementation of the service oriented architecture (SOA), they are based on a set of standards, SOAP, UDDI, WSDL, [9], and BPEL [15], which enable a flexible way for applications to interact with each other over networks. SOAP (Simple Object Access Protocol) is a protocol allowing applications to communicate with each other. UDDI (Universal Description Discovery Integration) defines a registry for service providers to publish their services. WSDL (Web Services Description Language) is used to describe a web service's capabilities and the interface to invoke it. A WSDL document is self-describing so that a service consumer can examine the functionality of the web service, at runtime and generate corresponding code to automatically invoke the service. The BPEL language is a workflow model used to represent the control flow and the data flow of a given business process.

In spite of these advantages, the infrastructure is not sufficient to ensure the automation of several tasks such as the service selection. In many cases, the industrials need to compose several services (or tasks) in order to fulfill a given requirement, each service is characterized by a set of QOS criteria (non-functional properties), and the whole composition is also characterized by several aggregated QOS values. In addition to that, the users impose a set of constraints or (end-to-end QOS requirements) like reputation, latency and cost (see the motivating scenario of the second section). The service composer must verify that the aggregated QOS values of the selected services match the user requirements at the start of the execution as well as during the run time. Nonetheless, dynamic fluctuations, such as failure (or QOS deviation) of certain provided services,(or changes in the user's QOS constraints), can cause the failure of the composition, Therefore, a quick adaptation to these change is crucial (such as the substitution of unavailable services). Moreover we must design a selection system that responses within a limited lapse of time, (so as to have a good performance).

It is worth noting that, this problem is NP Hard, in fact it is similar to the Multi-Choice Multidimensional Knapsack problem (MMKP) [16] which is well known in the domain of combinatory optimization. Consequently, the chance of finding an optimal solution in a reasonable time is very weak[13].

The goal of this paper is to build an efficient web service selection system that optimizes the aggregated QOS of the composition, and satisfies the end to end user's QOS constraints. Therefore we adopt a genetic based solution, which performs a global optimization, in the search space. Our choice is motivated by the aptitude of this meta-heuristic to handle large space of possible solutions. (By using a set of heuristics such as mutations and crossovers).

The rest of the paper is organized as follows: the section 2 presents a motivating scenario, the section3 reports a survey on the selection problem, the fourth section presents the problem modeling, in the fifth section we introduce the developed approach, the section 6 shows the obtained results and finally we present in the last section our conclusions and, we give the directions for future work.

II. THE MOTIVATING EXAMPLE

To illustrate the selection problem we consider a scenario [18] which involves a customer, who wants to buy a used car having a specific model, make, and mileage. The customer naturally aims to get the best deal.

We suppose that the customer can access several kinds of web services that play a role in the car purchase scenario. In this example we may access three kinds of services Car Purchase (CP), Car Insurance (CI), and FInancing (FI). A single Web service may provide multiple operations.

Different operations may also have dependency relationships. For example, the paymentHistory and financingQuote operations are both offered by the financing

service. The latter operation depends on the former operation, i.e., the payment history decides the financing quote. We also anticipate that there will be multiple competitors to provide each of the services mentioned above. It is important that the users' quality requirements be reflected in the service query as criteria for service selection. To purchase an entire car package, the customer would first like to know the price quote of the selected car and the vehicle history report. He then needs to get the insurance quote.

Finally, he also wants to know the financing quote. In addition, the client may have special constraints on the quality of the service operations. For example, he wants to spend less than 20 dollars to get the vehicle history report. Moreover, we can also express global constraints which handle the entire composition, for instance we may want to use only a whole composition which does not exceed 100 dollars).

The solution template of this composition involves 03 classes of services, each service is consumed by calling 02 operations.

Each operation is characterized by R QOS values, the global composition must fulfill R end to end constraints. Request=((op1,op2) ∈CP, (op3,op4) ∈CI, (op5,op6) ∈FI)

Each class or task can be fulfilled by a service instance Insi, which is characterized by a vector of QOS values, we can have L instances per class, and therefore we can have L^3 possible solutions. Finally we can resume this issue as follows:

We must search a composition c=(op1, op2, op3, op4, op5, op6) such that:

-U'(c) is maximized

-Each global constraint j is fulfilled by c (j=1..R)

U'(c): denotes the aggregated function of the different QOS attributes applied on c.

III. STATE OF THE ART

There are several approaches proposed in the domain of service selection. These approaches can be grouped in 02 major categories [4]: the multi-objective optimization and the mono-objective optimization. According to [18] we can adopt several database techniques to tackle the multi-objective selection, for instance we can use the divide and conquer algorithm, the bitmap algorithm, the index based algorithm, and the nearest neighbor algorithm.

The mono-objective class involves several approaches, [3,5, 6, 7,8,11, 12,20,21, 22]. In [12] the authors propose an extensible QoS computation model that supports open and fair management of QOS data. The problem of QOS-based composition is not addressed by this work.

The mono-objective category can use a global selection model [5, 6,14,21, 22] or a local selection model [11, 7] or a hybrid selection model [3]

The global selection model can determine the optimal solution, but it has an exponential complexity, however the local model has only a linear complexity but cannot handle the global constraints (there are only, local constraints). The third category is a compromise of the two approaches, it has a reduced complexity in comparison with the global approach, and it can also handle the global constraints.

The work of Zeng at al [21, 22] focuses on dynamic and quality-driven selection of services. The authors adopt global planning to find the best service components for the composition. They use (mixed) linear programming techniques [18] to find the optimal selection of component services. Similar to this approach Ardagna et, [5, 6] extends the linear programming model to include local constraints. Linear programming methods are very effective when the size of the problem is small. Nevertheless these methods suffer from weak scalability due to the exponential time complexity of the applied search algorithms [13].

In [20] the authors use heuristic algorithms that can be used to find a near-to-optimal solution more efficiently than exact solutions. The authors propose two models for the QoS-based service composition problem: 1) a combinatorial model and 2) a graph model. A heuristic algorithm is introduced for each model. The time complexity of the heuristic algorithm for the combinatorial model (WS HEU) is polynomial, whereas the complexity of the heuristic algorithm for the graph model (MCSP-K) is exponential. Despite the significant improvement of these algorithms, both algorithms do not scale with respect to an increasing number of web services and remain out of the real- time requirements. Any distributed implementation of these algorithms would raise a very high communication cost. The WS HEU for example, is an improvement of the original heuristic algorithm for solving general Multi-Choice Multi-dimensional Knapsack problems named M-HEU [1].

The use of the algorithm in a distributed setting, where the QOS values of the different service categories is managed by distributed service brokers would raise very high communication cost among these brokers to find the best composition. In this paper, we propose a mono-objective solution which is based on genetic algorithm. This later can solve the composition problem more efficiently and may be easily adapted to the multi-objective case.

IV. THE PROBLEM MODELING

A. The Composite Service

We assume that we have n abstract classes of web services. Each service class Sj ∈ S (e.g. car purchase services) is used to describe a set of functionally- equivalent web services .In this paper we assume that information about service classes is managed by a set of service brokers as described in [11, 12]. Web services can join and leave service classes at any time by means of a subscription mechanism. It is worth noting to distinguish between an abstraction composition (also called solution template) noted (S1,S2,…Sn) and concrete composition c (which composed of real instances) noted (Ins1,Ins2….Insn)

This later can be obtained by binding each abstract service class Si to a concrete web service Insj , such that Insj ∈ Si. We use c to denote a concrete composite service.

B. QOS Criteria

Our work considers only quantitative non-functional properties of web services, [21, 12]. These properties can include generic QOS attributes such as response time, availability, price, reputation etc, as well as domain-specific QOS attributes like bandwidth for multimedia web services as long as these attributes can be quantified and represented by real numbers.

We use the vector Q = {Q1(s), . . . , QR(s)} to represent the QOS attributes of a service s, where the function Qi(s) determines the value of the i-th quality attribute of s. The QOS values can be either collected from service providers directly (e.g. price), recorded from previous execution monitoring (e.g. response time) or from use feedbacks (e.g. reputation) [15]. The set of QOS attributes can be divided into two subsets: positive and negative QOS attributes. The values of positive attributes need to be maximized (e.g. reliability, availability…), however the value of negative attributes need to be minimized (e.g. price, response time). To homogenize these criteria, we convert the negative attributes into positive attributes by multiplying their values by -1.

C. The Aggregation Method

The QOS value of a composite service depends on the QOS values of its components as well as the composition model used (e.g. sequential, parallel, conditional and/or loops). In this paper, we consider only the sequential model. The Other models can be treated by using other Techniques [8].

The QOS vector for a composite service c is defined as QOS'(c) = {Q'1(c), . . . , Q'R(c)}. Q'i(c) represents the estimated value of the i-th QOS attribute of c and can be aggregated from the expected QOS values of its component services.

Our model uses three types of QOS aggregation functions (inspired from [18]): 1) summation (for the price, the reputation and the response time, 2) multiplication for the reliability and availability and 3) minimum/maximum relation for the values' normalization.

In this case study we set R to 5.

TABLE I. THE QOS AGGREGATION FUNCTIONS

QOS Criterion	**Agregation function**
Response Time	$Q1'(C) = \sum_{j=1}^{n} Q1(sj)$
Reputation	$Q2'(C) = 1/n * \sum_{j=1}^{n} Q2(sj)$
Price	$Q3'(C) = \sum_{j=1}^{n} Q3(sj)$
Reliability	$Q4'(C) = \Pi_{j=1}^{n} Q4(sj)$
Availability	$Q5'(C) = \Pi_{j=1}^{n} Q5(sj)$

D. The Global Constraints

The Global QOS constraints may be expressed in terms of upper and/or lower bounds for the aggregated values of the different QOS criteria. As mentioned earlier, we only consider positive QOS criteria. Therefore we have only lower bound constraints.

Let AC be an abstract composition and CONS= {cons1, ..$cons_m$…, $cons_R$ }, $0 \leq m \leq R$, be a vector of global constraints on AC (a vector of real values). Let c a concrete composition, in which a concrete web service is associated for each service class.

We say that c is feasible iff $QOS'(c) \geq CONS$, This means that all the global constraints are satisfied.

E. The Objective Function

In order to evaluation of a multi-dimensional quality of a service, we use a utility function that associates a real value to the QOS vector. In this work we adopt a mono-objective approach (ie a single objective function), this later is conceived as a Weighted sum of the different QOS values [19]. The utility computation involves a scaling process of the QOS attributes' values, to allow a uniform measurement of the multi-dimensional service qualities.

The scaling step is then followed by a weighting process which models the user priorities.

The scaling process of the QOS values gives a score comprised between 0 and 1.

Formally, the minimum and maximum aggregated values of the k-th QoS attribute of c are computed as follows:

$$Qmin'(k) = \sum_{j=1}^{n} Qmin(j, k) \quad \ldots\ldots\ldots (1)$$

$$Qmax'(k) = \sum_{j=1}^{n} Qmax(j, k)$$

$$Qmin(j, k) = \min_{\forall sji \in Sj} Qk(sji) \quad \ldots\ldots(2)$$

$$Qmax(j, k) = \max_{\forall sji \in Sj} Qk(sji)$$

Where Qmin(j, k) is the minimum value (e.g. minimum price) and Qmax(j, k) is the maximum value (e.g. maximum price) contained in the service class Sj.

The utility of a component web service $s \in Sj$ is computed as follows:

$$U(s) = \sum_{k=1}^{R} wk * (Qk(s) - Qmin(j, k))/(Qmax(j, k) - Qmin(j, k)) \quad \ldots.(3)$$

And the overall utility of a composite service c is computed as follows

$$U'(c) = \sum_{k=1}^{R} wk * ((Q'k(c) - Qmin'(k))/(Qmax'(k) - Qmin'(k)) \ldots (4)$$

with $wk \in R+$ and $\sum_{k=1}^{R}(wk) = 1$ are the weights (importance) of Q'k .

A concrete composition c is optimal if it is feasible and if it has the maximum value of the function U'.

We notice that the selection of the optimal concrete composition is NP hard (we must enumerate an exponential number of cases). The selection problem addressed here is formalized as follows:

Let CA = {S1, . . . , Sn} be a client request (or an abstract composition) And Cons a vector of m global QOS constraints {cons1,.. , consR}. We must find a concrete composition c={s1, . . . , sn} by binding each Sj to a concrete service sj' ∈ Sj such that:

1. U'(c) is maximized, and

2. Q' k(c) ≥ Cons(k), ∀ Cons(k)∈ CONS

V. THE PROPOSED APPROACH

The adopted genetic algorithm possesses the following attributes:

The population size=40 chromosomes, (the population is initialized randomly)

The maximum number of iterations=100

The crossover rate varies from 10% to 99%

The mutation rate varies from 1% to 20%

The objective function of the genetic algorithm 'f(x)', combines the function U'(x) and a penalty function p(x). p(x) decreases the utility score 'f(x)'of the solution that violates the global constraints.

Several penalty functions are proposed in the literature [17,2], (we have static, dynamic, adaptive.. functions), for the sake of simplicity we have chosen the following version [10] (a dynamic function):

$P(c)= -t*\sum_{k=1}^{R} (D_k)^2(c)$ Where

$D_k(c)= \begin{cases} 0 \text{ if } Q'k(c) \geq Cons(k) \\ |Q'k(c)- Cons(k)| \text{ otherwise} \end{cases}$

And t represents the current iteration number of the genetic algorithm.

This formula means that a rigorous penalty is applied when we get closer to the end of the optimization process. Finally the objective function is defined as follows:

f(x)=U'(x)+p(x)

VI. THE EXPERIMENTATION

In order to validate the approach, we consider a data set inspired from [18], we have 03 classes of services, and each class contains 100 instances, the total number of candidate solutions= 100^3.

The number of QOS attributes is fixed to 5. The QOS value of each attribute is generated by a uniform random process which respects the bounds specified in the following table.

All the criteria have the same priority (ie w_k=0.2)

TABLE II. THE SELECTION DATA SET

QOS	Car Purchase	Car Insurance	Financing
Response Time	0-300(s)	0-300(s)	0-300(s)
Reputation	0-5	0-5	0-5
Price	0-30($)	0-30($)	0-30($)
Reliability	0.5-1.0	0.5-1.0	0.5-1.0
Availability	0.7-1.0	0.7-1.0	0.7-1.0

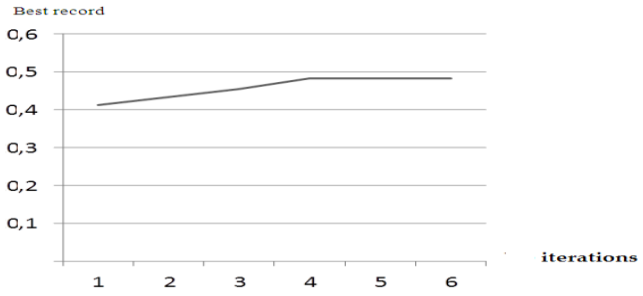

Figure 1. The score evolution of the best configuration

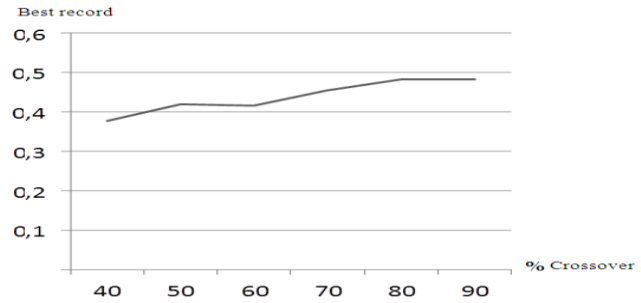

Figure 2. The score evolution according to the crossover (mutation=5%)

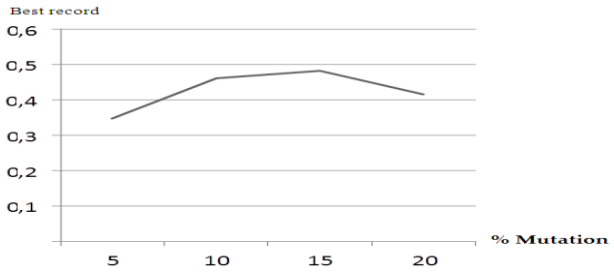

Figure 3. The score evolution according to the mutation (crossover=60%)

Our approach is more efficient than the mixed integer linear programming algorithm [14], because the adopted meta-heuristic is able to reduce the space search by applying a set of heuristics, such as mutations, crossover, penalty functions...etc.

The adopted penalty function has several advantages in comparison with other propositions, first of all, it increases the penalty as generation grows, (to ensure a feasible solution), it is also characterized by a reduced number of parameters in comparison with others approaches. As shown in the figure 1 the configuration that gives the best score of the utility function

f is (mutation=5%, crossover=80%), (we have done several simulations to get this result).

We notice that a stagnation situation is reached (see the figure 2) when the crossover exceeds 80%. A great value will only extend the space search, without having a significant improvement (the optimum, is usually located in a small region).

According to the third figure, we notice a performance deterioration if the mutation rate exceeds the interval [8 15], these high mutation's rates are caused by the weak value of the crossover factor.

VII. CONCLUSION

In this paper we have presented a genetic based approach for the selection of the best (near optimal) composition of services. The optimization process allows the specification of global constraints on the QOS attributes. The presented solution handles the global constraints by using a penalty function.

Future work will consider the use of the other meta-heuristics, such as swarm-particle optimization, Ants colony optimization..., Furthermore we may adopt a multi-objective optimization (the processing of several goals in the same time).

REFERENCES

[1] M. M. Akbar, E. G. Manning, G. C. Shoja, and S. Khan. Heuristic solutions for the multiple-choice multi-dimension knapsack problem. In Proceedings of the International Conference on Computational Science-Part II, pages 659–668, London, UK, 2001. Springer-Verlag.

[2] J. Adeli, H. and Cheng, N.T. Augmented lagrangian genetic algorithm for structural optimization, Journal of Aerospace Engineering, 7, 104-118, 1994.

[3] E.Alrifai , T. Risse Combining Global Optimization with Local election for Efficient QoS-aware Service Composition In WWW09, April 20–24, 2009, Madrid, Spain.

[4] E.Alrifai, T. Risse Selecting Skyline Services for QoS-based Web Service Composition In Proceedings of the WWW 2010, April 26–30, 2010, Raleigh, North Carolina, USA.

[5] D. Ardagna and B. Pernici. Global and local qos constraints guarantee in web service selection. In Proceedings of the IEEE International Conference on Web Services, pages 805–806, Washington, DC, USA, 2005. IEEE Computer Society.

[6] D. Ardagna and B. Pernici. Adaptive service composition in flexible processes. IEEE Transactions on Software Engineering, 33(6):369–384, 2007.

[7] B. Benatallah, Q. Z. Sheng, A. H. H. Ngu, and M. Dumas. Declarative composition and peer-to-peer provisioning of dynamic web services. In Proceedings of the International Conference on Data Engineering, pages 297–308, Washington, DC, USA, 2002. IEEE Computer Society.

[8] [8] J. Cardoso, J. Miller, A. Sheth, and J. Arnold. Quality of service for workflows and web service processes. Journal of Web Semantics, 1:281–308, 2004.

[9] F.Curbera, F.Duftler, R. Khalaf, W.Nagy, N. Mukhi, and S.Weerawarana .Unraveling . the Web Services Web: An Introduction to SOAP, WSDL, and UDDI. IEEE Internet Computing, 6(2). (2002).

[10] J. Joines, and C. Houck, On the use of non-stationary enalty functions to solve non-linear constrained optimization problems with GAs, Proceedings of the First IEEE International Conference on Evolutionary Computation, IEEE Press, 579- 584, 1994.

[11] F. Li, F. Yang, K. Shuang, and S. Su. Q-peer: A decentralized qos registry architecture for web services. In Proceedings of the International Conference on Services Computing, pages 145–156, 2007

[12] Y. Liu, A. H. H. Ngu, and L. Zeng. Qos computation and policing in dynamic web service selection. In Proceedings of the International World Wide Web Conference, pages 66–73, 2004

[13] I. Maros. Computational Techniques of the Simplex Method. Springer, 2003.

[14] G. L. Nemhauser and L. A. Wolsey. Integer and Combinatorial Optimization. Wiley-Interscience, New York, NY, USA, 1988.

[15] OASIS. Web services business process execution language, April 2007. http://docs.oasis-open.org/ wsbpel/2.0/wsbpel-v2.0.pdf.

[16] D. Pisinger. Algorithms for Knapsack Problems. PhD thesis, University of Copenhagen, Dept. of Computer Science, February 1995.

[17] O Yeniay penalty function methods for constrained optimization with genetic algorithms journal of Mathematical and Computational Applications, Vol. 10, No. 1, pp. 45-56, 2005.

[18] Q Yu, A Bouguettaya. Foundations for Efficient Web Service Selection Springer Science+Business Media, 2010.

[19] K. . P. Yoon and C.-L. Hwang. Multiple Attribute Decision Making: An Introduction (Quantitative Applications in the Social Sciences). Sage Publications, 1995

[20] T. Yu, Y. Zhang, and K.-J. Lin. Efficient algorithms for web services selection with end-to-end qos constraints. ACM Transactions on the Web, 1(1), 2007.

[21] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Z. Sheng. Quality driven web services composition. In Proceedings of the International World Wide Web Conference, pages 411–421, 2003.

[22] L. Zeng, B. Benatallah, A. H. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang. Qos-aware middleware for web services composition. IEEE Transactions on Software Engineering, 30(5):311–327, 2004.