Hybrid Web Service Selection based on Functional and Non-Functional Properties

Halfaoui Amal
Computer science department
Tlemcen university-Algeria
a_halfaoui@mail.univ-tlemcen.dz

Hadjila Fethallah
Computer science department
Tlemcen university-Algeria
f_hadjila@mail.univ-tlemcen.dz

Didi Fedoua
Computer science department
Tlemcen university-Algeria
f_didi@mail.univ-tlemcen.dz

Abstract - Web service composition enables seamless and dynamic integration of business application on the web. The performance of the composed application is determined by the performance of the involved web services. Therefore, the selection of composite services is a very complex and challenging task, it becomes a decision problem on which component services should be selected so that user's preferences are satisfied, especially when it has to consider not only the user's QoS preferences (non-functional aspect) but also user's functional Preferences (functional aspect). In this paper we address this problem and present a solution that combines local selection with global optimization. The proposed solution consists of two steps: First, in the local selection, we take into account both of users' preferences and QoS requirements and use dominance relationship to select the top-k services and reduce the services involved in the composition. Second, we use a global optimization to fulfil the global requirement and select the Optimal composition by using the Tabu Search Algorithm. The experimental evaluation indicates that our approach significantly outperforms existing solutions in terms of optimality.

Keywords - Web Services Selection, Optimization, Services Composition.

1 INTRODUCTION

The Selection of an appropriate Web service for a particular task has become a difficult challenge due to the increasing number of Web services offering similar functionalities. The functional propertie mtd mmmzmzmb s describe what the service can do and the non-functional properties depict how the service can do it. The requirements of the user are rarely satisfied by one web service but generally we need more than one. The web service com-position consists in building a value-added ser-vices and web applications by integrating and composing existing elementary web services. A lot of approaches have been proposed, they include AI planning techniques [15], formal models [7] (finite states machines, petri nets,...) and meta-heuristics[8][19]. The majority of them does not address the functional, the non functional aspects and the global constraints in the same time. Our purpose is to take into account both of the two aspects(functional, n-fonctional) and fulfill the global constraints. In order to explain our motivations let us consider the following example:

The example [6] in Table-1 presents an e-commerce system supporting users to buy cars. Each service has its input i() and output o() parameters. Services providing the same functionality belong to the same class(S_{21}, S_{22}, S_{23} belong to S_2). Each service has:

i. Functional constraints on the data it manipulates, For instance the cars returned by S_{22} have prices between 5500 and 7000 euro.

ii. N-functional constraints, Services set their values associated to three parameters q1,q2,q3 corresponding to price, response time and availability. Suppose that a user x wants to buy a french car preferably at an affordable price[5000,7000] with warranty between 12 and 18 months, having a power [60,80] and consumption [10,11]. The user preference concern also the global constraints, like the total price of the composition don't exceed 10 euro.

The user will have to invoke S_{11}, he can then invoke

Table 1: Example of Web Services.

service	functionality	functional constraints	N-Functional		
			q_1 ()	q_2 (ms)	q_3 (ms)
$S_{11}(i(x), o(y)$	Return the automakers y made in x	-	8	120	70
$S_{21}(i(x), o(y,z,t))$	Return the car y along with their prices z and waranties t for a given automaker x	z is [5000,7000], t is [12,24]	1	140	50
$S_{22}(i(x), o(y,z,t))$		z is [5500,7000],t is [12,18]	1	145	35
$S_{23}(i(x), o(y,z,t))$		z is [15000,20000] t is [6,24]	3	160	30
$S_{31}(i(x), o(y,z))$	Return the power y and the concumption z for a given car x	y is [50,70], z is [4,10]	3	170	40
$S_{32}(i(x), o(y,z))$		y is [50,90], z is [8,12]	8	150	50
$S_{33}(i(x), o(y,z))$		y is [200,400], z is[10,20]	4	160	30

one or more of services belongs to class S_2 finally, he invoke one or more of services belongs to class S_3.
How to find the optimal composition c that satisfy:
i. The user preferences (functional Aspect) and at the same time satisfy (non functional Aspect) i.e the services involved in the composition c must maximize the positive QoS Criteria such as reliability and availability, and minimize the negative criteria such as cost and execution time.
ii. The global constraints which relate to the QoS attributes of the composition.
To face theses challenges
- We compute the matching degrees between services' constraints and user's preferences and in order to select the most relevants services and reduce space search, we use the concept of Fdominance relationship to select the top-k services of each class that will be involved in the composition process.
- We satisfy the global constraints by using the tabu search algorithm which has been successfully applied to a wide variety of problems because of it's simplicity and practical effectiveness.
Formally our proposal can be described as follows:
i. we have to find the top-k services S_{ij} in each class S_i that match the user preferences QoS by using the matching degrees M and ranking them by considering both of QoS and Functional constraints by using the Fdominance score.
ii. we have to search a composition c=(s1, ..., sn) such that we have to maximise the function U(c) and satisfy each global constraint, to this end we use the Tabu Search Algorithm. U(c): denotes the Fdominance Score of the functional and n-functional properties of c. The Fdominance Score is detailed further.
The main contributions of this paper are summarized as follow :
1.We consider both of the functional and the non-functional user preferences at the same level. We notice that the best services which fulfill the functional properties are not always those which maximise the QoS properties and vice versa, so considering the two properties is a compromise to match as possible the two aspects at the same time.
2.We Combine the local and the global optimization. Since the number of candidate services for a composition may still too large, we use the FDominance relationship to reduce the search space and select the Top-k services. After that we use the Tabu Search for the global optimization to satisfy global constraints. We notice that the best service in each class produces the best composition, this one is not the best for the user if it violates the global constraints.
The rest of the paper is organized as follows. In the next section we discuss related work. In Section 3, we formalize the problem and present our approach. Section 4 presents performance analysis and experimentations. Finally, section 5 gives conclusions and an outlook on possible continuations of our work.

2 STATE OF THE ART

A lot of efforts have been devoted to the service selection/composition. We distinguish three major categories: the selection based on QoS, the selection based on functional preferences and the selection based on QoS and functional preferences.
The first category takes only the user preferences into account to rank candidate services. Many searches have been made in this domain, some of them uses the Pareto dominance [18] [17], the others use the fuzzy set theory to models preferences to select top k dominant skylines [1] [5] [6].
The second category is based only on the QoS parameters, it can use two types of approaches [2] the multi-objective optimization and the mono-objective optimization. The multi-objective selection can be supported by using database techniques like the divide and conquer algorithm, the bitmap algorithm, the index based algorithm(b-tree, hash table) and the nearest neighbor algorithm (R tree). The mono-objective class involves several approaches [4][9] [16] [23], which can use global

selection pattern[4] [23] or local selection pattern or hybrid selection pattern [11]
The third category is based on the QoS constraints (non functional aspect) and Users preference constraints(functional aspect). The majority of research does not address the functional and the non functional aspects in the same time, especially for the workflow based composition. Our contribution belongs to this category.

3 THE PROBLEM FORMULATION

Our proposed approach contains two phases. Phase 1 (The local selection): we compute the top k services and reduce the search space by using the FDominance. Phase2 (The global optimization): We find the near optimal composition that satisfy the global constraints by using Tabu search Algorithm.

3.1 The local selection

Let be Rq a user's request where $Rq = \{fc, gc\}$, fc is a vector of user preferences $fc = \{fc_1, .., fc_m\}$ and $gc = \{gc_1, gc_2, .., gc_n\}$ is a vector of n QoS global constraints like time, reputation.
Given a set of services classes $S = \{S_1, .., S_n\}$ where a class S_j regroups the services that had the same functionalities but different constraints.
Let be S_{ij} the j-th service of the class i. It is described as a set of two vectors : vector Q for Q_oS attributes and Vector f for Functional constraints. we use the vector fc_i to represent the subset of fc that corresponds to the constraints involved in the services of the class Si.
The request of the example is
Rq=fc(fc_1=(),fc_2=($[5000, 7000], [12, 18]$),fc_3=($[60, 80]$,$[10, 11]$)),gc($q1 < 10$).

3.1.1 The Functional Constraints

The service selection process starts with computing the matching degree between the request Rq and services S_{ij}.
Given the users preferences on service description attributes fc, the degrees of match between a requested Rq and an available service (see e.g.,[10]) are computed. In this work, we use the Jaccard coefficient for matching service descriptions. If $I1, I2$ are two intervals, their Jaccard coefficient is $J(I1, I2) = \frac{|I1 \cap I2|}{|I1 \cup I2|}$, where $|I|$ measures the length of the interval.
We suppose that we have m functional constraint $f(S_{ij}) = \{f_1(S_{ij}), .., f_m(S_{ij})\}$ where $f_i(S_{ij})$ is the value of the i-th functional constraint of the service S_{ij}.
We use the Vector $M(cf_i, S_{ij}) = (m_1(S_{ij}), .., m_n(S_{ij}))$ to represent the matching degrees between $f(S_{ij})$ and the subset vector of preference constraints fc_i of the user's query Rq where the function $m_k(S_{ij})$ is the value of matching degrees (the Jaccard coefficient) between the k -th functional constraint offc_i and the k-th functional constraint of $f_i(S_{ij})$.The table 2 show the Jaccard value macthing of the example 1

Table** 2: **Matching degrees between The query and services

	fc_i	S_{ij}	$M(fc_i, f(S_{ij}))$	
			m_1	m_2
Rq	fc_1	S_{11}	-	-
	fc_2	S_{21}	1	$0,5$
		S_{22}	0.75	1
		S_{23}	0	0.33
	fc_3	S_{31}	0.6	0.14
		S_{32}	0.50	0.25
		S_{33}	0	0.1

3.1.2 Non-functional Constraints (QoS))

We suppose that we have R quantitative Q_oS values for a service S_{ij}. we use the vector $Q(S_{ij}) = \{Nq_1(S_{ij}), .., Nq_r(Sij)\}$ to represent the Q_oS attributes of a service S_{ij} where the function $Nq_k(S_{ij})$ represent the k-th Normalized quality attribute of S_{ij}. We convert the negative attributes(time, cost) into positive attributes by multiplying their values by -1 such that the higher value is the higher quality. To allow for a uniform measurement of Web service qualities independent of units, we normalize the different QoS values in the range [0, 1], as follow :

$$Nq_k(S_{ij}) = \frac{q_k(S_{ij}) - Qmin(q_k)}{Qmax(q_k) - Qmin(q_k)} \quad (1)$$

Where $Nq_k(S_{ij})$ is the normalized QoS value of the Web service S_{ij} on the QoS parameter q_k and $Qmin(q_k)$ (resp.$Qmax(q_k)$ is the minimum (resp. maximum) value of the QoS parameter q_k. Table3 is the extention of the table 2 with the QoS values of Web services example of Table 1 after normalization.

3.1.3 The top-k relevant services

Let us consider the services in Table 3, to compute the Top-K services by taking into account the

Table 3: Web services with Matching score and Normalized QoS values

	fc_i	S_{ij}	$M(fc_i, f(S_{ij}))$		$Q(S_i)$		
			m_1	m_2	Nq_1	Nq_2	Nq_3
Rq	fc_1	S_{11}	-	-	1	1	1
	fc_2	S_{21}	1	0,5	1	1	0
		S_{22}	0.75	1	1	0.75	0.75
		S_{23}	0	0.33	0	0	1
	fc_3	S_{31}	0.6	0.14	1	1	0.5
		S_{32}	0.50	0.25	0	0	0
		S_{33}	0	0.1	0.8	0.5	1

functional and non- functional parameters,we have to compare the services of the same class by considering the vectors $(M(), Q())$ of each service S_{ij} (see Table 3) .

Definition1 (Service Dominance) A Web service S_{ij} is said to dominate (Pareto dominance) another Web service S_{ik} if and only if S_{ij} is better than or equal to S_{ik} in all parameters and better than S_{ik} in at least on one parameter.

Definition2 (User Preferences-aware Service Dominance) Given a user Functional-preference space M, and a user non-Functional space Q, a service S_{ij} is said to dominate an other service S_{ik} on M and Q if and only if $\forall m_i \in M$, $\forall Nq_i \in Q$, $(m_i(S_{ij}) \geq m_i(S_{ik})) \wedge (Nq_i(S_{ij}) \geq Nq_i(S_{ik}))$ and $\exists Nq_t \in Q$, $\exists m_t \in M$, $(m_t(S_{ij}) > m_i(S_{ik}))\wedge$ $(q_t(S_{ij}) > q_i(S_{ik}))$
according to our example table 3 we have S_{21} dominates S_{23}, let consider now S_{21} and S_{22}, in fact neither S_{21} dominates S_{22} nor S_{22} dominates S_{21} because S_{21} is better than S_{22} in $m1$ and q_2, and S_{22} is better than S_{21} in m_2 and q_3, so S_{22} and S_{22} are incomparable. However we can consider that S_{22} is better than S_{21} since $q_3(S_{22}) = 0.75$ is much higher than $q_3(S_{21}) = 0$, and $q_2(S_{22}) = 0.75$ is almost close to $q_2(S_{21} = 1)$. here for, it is more interesting to use the FuzzyDominance score(FDominance) relationship defined in [5] to express the extent to which a matching degree (more or less) dominates another one.

Definition3 (FDominance) given two d-dimentional point u and v the Fdominance express the extent to which u dominates v as

$$deg(u > v) = \frac{\sum_{i=1}^{d} \mu \gg (u_i, v_i)}{d} \tag{2}$$

Where $\mu >> (u_i, v_i)$ expresses,the extent to which u_i is more or less greater than vi it's defined as

$$\mu \gg (u_i, v_i) = \begin{cases} 0 & \text{if } x - y \leq \varepsilon \\ 1 & \text{if } x - y \geq \lambda + \varepsilon \\ \frac{x-y-\varepsilon}{\lambda} & \text{otherwise} \end{cases} \tag{3}$$

$$FDS(S_{ij}) = \frac{1}{S_i - 1} \sum_{S_{ik} \in S_i} deg(S_{ij} > S_{ik}) \tag{4}$$

Let's return to our example $deg(S_{21} > S_{22}) = 0.4$ and $deg(S_{22} > S_{21}) = 0.29$ with $\epsilon = 0.1$ and $\lambda = 0.2$. This is more significative than S_{21} and S_{22} are not comparable.
We compute $FDS(S_{ij})$ of all services for each class in order to rank them. After that we take the Top-K services. Only the Top-k services will be considered in the global optimization step.

3.2 The Global Optimization

3.2.1 The Global QoS Constraints

Let $C = \{S_{1i_1}, .., S_{ni_n}\}$ be a composition of n services. The global Q_oS constraints gc may be expressed in term of upper and/or lower bound for the aggregated values of the different Q_oS criteria. we only consider positive Q_oS criteria to have only lower bound constraint. We say that a composition C is feasible if all the request's global constraints are satisfied, this means that $Qc(c) \geq gc$. where $Qc(c)$ is the vector of the Q_oS value of a composite service c.
The Q_oS value of a composite service depends on the Q_oS values of its components as well as the composition model used. In this paper we consider the sequential composition. The Q_oS vector of composite service C is defined as $Qc(C) = \{Qc_1(C), ..Qc_i)$. Where $Qc_i(c)\}$ represents the value of the i-th Q_oS attribute of C and can be aggregated from the Q_oS values of its components services by using the aggregation function inspired from [22], see Table 4.

Table 4: The Q_oS Aggregation functions

Q_oS	Agregation function
Reponse Time	$Qc_1(C) = \sum_{j=1}^{n} q_1(S_j)$
Reputation	$Qc_2(C) = 1/n * \sum_{j=1}^{n} q_2(S_j)$
Price	$Qc_3(C) = \sum_{j=1}^{n} q_3(S_j)$
Reliability	$Qc_4(C) = \Pi_{j=1}^{n} q_4(S_j)$
Availability	$Qc_5(C) = \Pi_{j=1}^{n} q_5(S_j)$

3.2.2 *The Utility function*

Different service composition can be generated from different S_i Top-K service Classes to answer a user query. In order to evaluate the service compositions, we need an objective function $u(c)$ that associates a value to the composition.

$$u(c) = FDS(c) + P(c); \tag{5}$$

where $FDS(c)$ is the FDominance(see definition3) Score associate to a composition $C = \{S_{1i_1}, .., S_{ni_n}\}$ as follows

$$u(c) = \frac{1}{n}\sum_{i=1}^{n} FDS(S_{ij_i}) \tag{6}$$

The function $p(c)$ is the penality function which decreases the utility score u(c) of the composition that violates the global constraints gc, it's defined as follow $P(c) = -\sum_{i=1}^{n} D_k(c)$ where

$$D_k = \begin{cases} 0 & \text{if } Q_k(c) \leq \ Gc_k \\ |Q_k(c) - Gck| & \text{otherwise} \end{cases} \tag{7}$$

3.2.3 *The Tabu-Search optimization approach*

In this paper we apply metaheuristic optimization techniques to select the optimal or a near-optimal solution in Web service composition, we use the Tabu search-based method and propose The TSOptiSelection Algorithm to maximize the objective function $u(c)$. Our algorithm (Algorithm1) uses a Tabu Search algorithm with diversification strategy. The Tabu search meta-heuristic relies on the principles of forbidding a set of elements which are stored in tabu list. The basic concepts defined in Tabu search are: The initial solution , neighborhood of a solution , tabu-active elements , tabu list, the objective function and diversification strategy.

a- The initial Solution
In our approach, the initial solution C_0 is a random composition, we choose a random service from each class, for an efficient initial solution the services should not have the same rank in order to have different composition by moves services between classes
let $Co = \{S_{1j_1}, .., S_{nj_n}\}$ be an initial solution, $C0$ is an efficient initial solution iff $\forall S_{ij_k} \in Co, \forall S_{mj_l} \in Co / i \neq m, k \neq l$

b- The neighborhood
The search space of our algorithm is the set of compositions obtained by moves applied to the initial solution. A move can be defined as a permutation between two services rank of two classes. A solution $c^{'} \in X$ is a neighbor solution $c \in X$ if it can be obtained by applying a move to c. Let's consider $C = \{S_{11}, S_{23}, S_{34}\}$, $C^{'} = \{S_{13}, S_{21}, S_{34}\}$ in this example, we replace the service s3 by the service s1 into the class 1 and change the service s1 by the service s3 into the class 2, we apply the move: permutation of the rank 1 and 3.
The Tabu list in our approach is the structure that contains the service composition used in the previous iterations. We use the objective function u(c) (see formula 1) to evaluate each neighbor solution.
We define the diversification strategy as restarting the process after a certain number of iterations when the Optimal local solution stagnates.

c- The Tabu-Search optimization algorithm
The algorithm, TSOptSelection, selects the Optimal or near optimal services composition according to the objective Function $u()$. The algorithm proceeds as follows.

Step.1 *Process the neighborhood of a solution compoition* (line 6-13).
For each solution, we generate all the neighbor solutions by using SwapMoves() that defines all the possible moves, then we compute the score of all the composition solutions in Ls (line 8) and choose the best neighbor of Ls (line 10). The best neighbor is the one having the highest score. if the best neighbor solution is in the tabu list, we choose the next Best solution that not in the tabu list (line 12). This one will be used in the next iteration.

Step.2 *Update the Tabu list and the optimal Solution* (line 14).
Add the best neighbor solution to the tabu list (avoid a cycle)

Step.3 *Restart the processing* (line18-19).
if we reach p% of iteration and we have no amelioration of the optimality we restart the processing Algorithm.
The algorithm ends when it satisfies the stop condition (reach the number of iterations or the stagnation of the solution for it iterations)

Algorithm 1 TSOptiSelection

```
Input: $Top\text{-}kRelevantsServices, t, it, p$ // t is the lenght of Tabulist,it is the number of iterations, p is the stagnation rate
Output: $C^*$ the optimal composition
1: $C^0 \leftarrow ComposeServices(random(S_{ij}), ..random(Sij))$ // an initial solution
2: $C^* \leftarrow C^0$
3: $Ls \leftarrow \{C^*\}$ // set of composition solution
4: $LTabu \leftarrow \{\}$ // Set of composition Tabu
5: while $NotStopCondition$ do
6:     $Ls \leftarrow SwapMoves(C^0)$ // Generate all the Neighborhood solutions of $C^*$
7:     for all $Cs$ in $Ls$ do
8:         $score = ComputeUtililyScore(C)$ // use the objectif function u(c)
9:     end for
10:    $C^0 \leftarrow MaxScoreComposition(lS)$
11:    while $C^0$ is Tabu do
12:        $C^0 \leftarrow NextMaxScoreComposition(lS)$
13:    end while
14:    $LTabu \leftarrow LTabu \cup C^0$
15:    if $Score(C^0) > Score(C^*)$ then
16:        $C^* \leftarrow C^0$
17:    end if
18:    if it = it*p and stagnation and Restart number $<$ threshold then
19:        TTSOptiSelection($Top\text{-}k$)
20:    end if
21: end while
22: return $C^*$
```

4 EXPERIMENTAL EVALUATION

This section briefly reports our experiments related to our approach. For this purpose we use a data set by assigning arbitrary values to 2000 services , we have 10 classes of services and each class contains 200 instances,each service has 02 functional-constraints and 5 QoS attributes. The QoS value of each attribute is generated by a uniform random process which respects the bound specified as follow: Response time (0-300s), Reputation (0-5), Price(0-30), Reliability(0.5-1.0), Availability(0.7-1.0).
several parameters have been modified to find the new optimal result in term of optimality
i. The maximum number of iteration: is [100,10000].
ii. The size of the Tabu List [7, 20].
iii. The stagnation rate P and the threshold of restart number.
All the experiments are taken on the same software and hardware, which were Intel i3-2365M CPU @ 1.40GHz 4 processors, 4.0GB of RAM, Ubuntu 13.10, Netbeans 7.4. several simulations have been made to compare our Approach:hybrid-Optimization(Tabu Search with Top-K).) to the Standard Tabu Search. For simplicity we denote our approach (TS (Top-K)).

We compare the optimality rate of TS (Top-K) and the standard Tabu Search . We consider several top k (top 100, top 50, top 20 and top 10) The optimality Score is defined as follows: rate=the fitness of the current solution/the fitness of the optimal solution. The optimal solution's fitness for this base, is equal to 0.62, therefore the optimality rate of a solution 'a' is u(a)/0.62
As depicted in the Figure 1 the use of the top-k se-

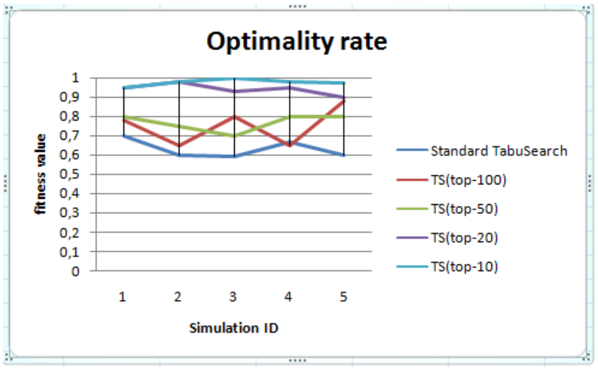

Figure** 1: **The optimality rates of the Tabu Search/ TS(Top-K) with constraints(it=500,p=0.2,t=10)

lection combined with the tabu search largely outperforms the standard tabu search in term of optimality rate for all simulations, The main reason why the optimality rate of our aproach is better than the standard tabu search is the use of the Top-K services which reduces the search space of services composition and consider only the best services in each class.

5 CONCLUSION

In this paper, we have presented an optimization approach for web services composition. this latter takes into account the functional and n-functional properties at the same time. Our approach reduces the search space and handles the global constraints. Experimental results show that the proposed approach is effective and efficient. For future work, we will consider alternative algorithm like Particle Swarm Optimization with the use of dominance relation ship for efficient and fast Web Service Composition.

6. REFERENCES

[1] S. Agarwal and S. Lamparter. User preference based automated selection of web service compositions. *K. V. A. S. M. Z. C Bussler, editor, ICSOC Workshop on Dynamic Web Processes*, page 112, Decembre 2005.

[2] E. Alrifai and T. Risse. Selecting skyline services for qos-based web service composition. In *in Proceedings of the WWW*, pages 26–30, Raleigh, North Carolina, U SA., April 2010.

[3] M. Alrifai, D. Skoutas, and T. Risse. Selecting skyline services for qos-based web service composition. *WWW*, page 1120, 2010.

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

[5] K. Benouaret and D. Benslimane. On the use of fuzzy dominance for computing service skyline based on qos. *ICWS*, 2011.

[6] K. Benouaret, D. Benslimane, and A. Hadjali. A fuzzy framework for selecting top-k web service compositions. *Applied Computing Review*, 2011.

[7] D. Berardi, D. Calvanese, G. De Giacomo, R. Hull, and M .M ecella. Automatic composition of transition-based semantic web services with messaging. In ACM, editor, *31st VLDB Conference on Very Large Data Bases (VLDB 05)*, page 613624, 2005 Tronheim, Norway.

[8] Steffen Bleul, ThomasWeise, and Kurt Geihs. Making a fast semantic service composition system faster. In *In Proceedings of IEEE Joint Conference (CEC/EEE 2007) on E-Commerce Technology (9th CEC07) and Enterprise Computing, E-Commerce and E-Services (4th EEE07)*, 2007.

[9] J. Cardoso, J. M iller, A. Sheth, and J. Arnold. Quality of service for workflows and web service processes. *Journal of Web Semantics*, pages 281–308, 2004.

[10] X. Dong, A. Y. Halevy, J. Madhavan, E. Nemes, and J. Zhang. Simlarity search for web services. *in VLDB*, page 372383., 2004.

[11] E.Alrifai and T. Risse. Combining global optimization with local election for efficient qos-aware service composition. In *WWW09*, April 2024, 2009, M adrid, Spain.

[12] F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, 101 Philip Drive, Assinippi Park, Norwell, Massachusetts 02061, USA, 1998.

[13] F. Hadjila, Chikh A, and A Belabed. Semantic web service composition: a similarity measure based approach algorithm. In *ICIST11 Tebessa Algeria*, 2011.

[14] F. Hadjila, Chikh A, M. Merzoug, and Z Kameche. Qos-aware service selection based on swarm particle optimizatio. In *IEEE ICITES12*, 2012.

[15] F. Hadjila, A.Chikh, and A.Belabed. Semantic web service composition: a similarity m easure based approach algorithm. In *ICIST11 Tebessa Algeria*, 2011.

[16] F. Hadjila, A.Chikh, and M.Dali Yahia. Qos-aware service selection based on genetic algorithm. In *CIIA*, 2011.

[17] D. Skoutas. Ranking and clustering web services using multicriteria dominance relationships. *IEEE*, 2010.

[18] D. Skoutas, D. Sacharidis, A. Simitsis, V. Kantere, and T. K. Sellis. Top- dominant web services under multi-criteria matching. *EDBT*, page 898909, 2009.

[19] Weise T, Steffen B, and Kurt G. Web service composition systems for the web service challenge a detailed review. Technical report, A technical report number: urn:nbn:de:hebis:34- 2007111919638 university of kassel., 2007.

[20] H. Wang, J. Xu, , and P. Li. Incomplete preference-driven web service selection. *IEEE SCC*, 2011.

[21] Q. Wu, A. Iyengar, R. Subramanian, I. Rouvellou, I. Silva-Lepe, and T. A. Mikalsen. Combining quality of service and social information for ranking services. *ICSOC/ServiceWave*, page 561575, 2009.

[22] Q. Yu and A. Bouguettaya. Fondations for efficient web service selection. *Springer Science+Business Media, 2010.*, page 481488, 2010.

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