International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
Volume 3– No.6, July 2012 – www.ijais.org

Optimizing Semantic Web Service Composition by using Boid Particle Optimization

Hadjila Fethallah
Computer Sciences Department
Tlemcen university
B.P 119 Faculty of Sciences
Tlemcen Algeria

Chikh Mohammed Amine
Computer Sciences Department
Tlemcen university
B.P 119 Faculty of Sciences
Tlemcen Algeria

Merzoug Mohammed
Computer Sciences Department
Tlemcen university
B.P 119 Faculty of Sciences
Tlemcen Algeria

ABSTRACT
The Quality of Service (QoS) is a key factor in Web service selection and composition. In this paper, we propose a meta-heuristic based on renolds'Boid model, in order to compose a sequence of services that optimizes the QOS attributes, and conserves the semantic interaction between components, in addition to the global constraints required by the user. This approach uses multiple moving operators such as the cohesion, the alignment, the random velocity, and the social exchange of positions. This paper provides also an experimentation that evaluates the optimality rates of the approach.

General Terms
Web Technologies, Combinatorial Optimization.

Keywords
Service oriented architecture; boid particle optimization;qualiy of service; service composition,combinatory optimization;ontologies

1. INTRODUCTION
The Web service technology is immerging as a powerful vehicle for organizations that participate in Web based dynamic collaborations. Several standards, such as SOAP, UDDI, WSDL, [1], and BPEL [2], have been created to support the effective deployment of web services.

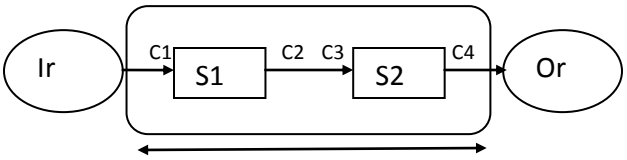
The Web service composition is one of the most important challenges in the service oriented architecture, it consist in building a value-added services and web applications by integrating and composing existing elementary web services. A lot of efforts have been proposed, in order to address this problem. Existing approaches advanced in the literature include AI planning techniques [3], formal models [4] (finite states machines, petri nets,…) and meta-heuristics[5,6]. The majority of them does not address the functional and the non functional aspects in the same time.

With the increasing number of web services with the same function on the Internet, a great amount of candidate services emerge. How to find a service according to the Qos requirements of users has become a hot issue in the service oriented architecture. In fact the users, are obliged to find a particular service from the candidate service set (or class), making the entire Qos of composite service best to meet users' need. Numerous researchers have been studying this problem from different perspectives (see the second section)

In order to explain the QOS aware composition problem, we consider the following example: we have a request composed of an input concept Ir, and an output concept Or , the user must search a composition c that accepts Ir as an input and gives Or as an ouput, in addition to that, the solution c must preserve the semantic coherence, ie there are no semantic conflicts between the components of the solution, for instance the output of S1 'c2' must be compatible with the input of S2 'c3'.

furthermore c must match the user's request (see the functions U2,U3).

Moreover c has to maximize the positive Qos Criteria such as reliability and availability, and minimize the negative criteria such as cost and execution time. In addition to that, we must also satisfy the global constraints which are related to the Qos attributes (we have R Qos Attributes), for instance, the cost of a solution c must be lesser than 1000 Euros. It is worth noting that, this problem is NP Hard[7].



The composition c

Fig 1: The motivating example

Formally the problem can be described as follows: we have to search a composition c=(s1, ..., sn) such that:

U1(c) is maximized

U2(c,request) is maximized

U3(c,request) is maximized

U4(c) is maximized

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

U1(c): denotes the aggregated function of the different QOS attributes applied on c.

U2(c,request): denotes the semantic closeness (or compatibility) between the composition inputs and the request inputs, in our example U2(c,request)=closeness (Ir,C1)

U3(c,request): denotes the semantic closeness (or compatibility) between the composition outputs and the request outputs, in our example U3(c,request)=closeness (Or,C4)

U4(c): denotes the semantic coherence of the composition c , in our example U4(c)=closeness (C2,C3)

The function "closeness " is detailed further.

The main contribution of this paper is to propose a mono objective optimization algorithm that takes into account the functional aspects (the functions U2 U3 U4) and the non functional aspects (the functions U1 and the global constraints) simultaneously. In comparison with our previous works [8,9] that consider only the non functional aspects, we modify the objective function to handle the functional aspect.

To tackle this problem, we adopt the Boid particle optimization approach (BSO) [10],

In 1986, Reynolds proposed an automated model of coordinated animal motion such as bird flocks and fish schools, and called the flocking creatures as boids. Each boid can make three simple steering behaviors which describe how an individual boid maneuvers based on the positions and velocities its nearby flockmates:

Separation : each boid tends to avoid crowding with its neighbors.

Alignment : each boid tends to be closer to the average heading of its neighbors.

Cohesion (Fig.3): each boid tends to be closer to the average position of its neighbors.

Our modified BSO algorithm can explore efficiently a large space solutions by using four moving operators : the cohesion moving, the alignment moving, the random moving, and the social moving.

These moving operators give the ability to avoid the local optimums, but they can increase the execution time of the optimization process.

Compared with the swarm particle optimization , BSO is closer to the biological behavior of the flocks, furthermore it is more immune to the local optimums than the SPO model[11].

The rest of the paper is organized as follows: the section 2 reports a survey on the composition and the selection problem, the third section presents the problem modeling, in the fourth section we introduce the developed approach, the fifth section shows the obtained results and finally, conclusions and future work are described in the last section.

2. RELATED WORKS

The field of service selection and composition has received considerable attention in recent years. Roughly speaking, We distinguish two types of approaches [12]:

the multi-objective optimization and the mono-objective optimization. (See the figure 1), the majority of them does not address the semantic aspect during the selection.

The multi-objective selection, (also called the skyline based optimization) can be handled by using multiple database techniques [13]. More specifically we can use the divide and conquer algorithm, the bitmap algorithm, the index based algorithm (B tree, hash table), and the nearest neighbor algorithm (R tree).

Furthermore, there are several works which takes into account the user preferences to select the top k dominant skylines [12,13] some of them uses the fuzzy set theory to model the preferences and the dominance relationship, the others uses the pareto-dominance concept to rank the web services.

The mono-objective class involves several approaches, [14,15,16,17,18, 8,19,20, 21].

This category can use a global selection pattern [18,15, 16,22,20,21] or a local selection pattern or a hybrid selection pattern[14].

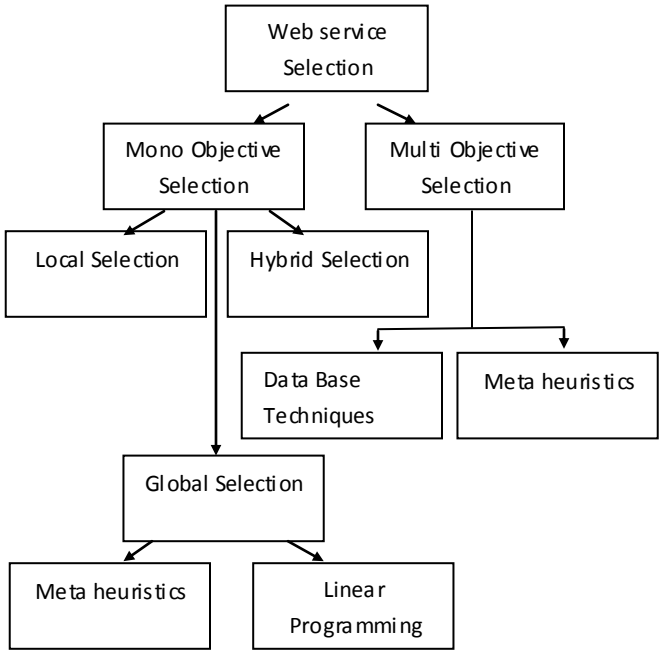


Fig 2: The selection approaches

The global selection model focuses on the entire composition (ie the n components of the solution), it can get the optimal solution, but it has an exponential complexity, nevertheless, the local model has only a linear complexity but cannot handle the global constraints (it handles only, local constraints).

The third category is a compromise of the two approaches, it begins the search with a global optimization, then we continue the work with a local one, its temporal complexity is lower than the global optimization, in addition to that it can also handle the global constraints.

In [8], the authors adopt a genetic algorithm (global selection) to select a near-optimal composition, The obtained results are very satisfactory, but the response time is too high.

Zeng and al [20, 21] employ the mixed linear programming techniques [23] to find the optimal selection of component services.

Similar to this approach Ardagna and al, [15, 16] modify the linear programming schema to include local constraints.

Linear programming methods are very strong when the size of the problem is small. but these methods are non scalable, because they have an exponential time complexity [23].

In this paper we propose a mono-objective approach, and more specifically a global selection approach. We have chosen the Boid particle optimization, because it is more suitable to NP Hard problems.

In addition to that, the algorithm has only a polynomial complexity and a lot of flexibility (the number of groups, the number of particles, the topology…).

3. THE PROBLEM MODELING

3.1 The QOS Modeling

Our QOS criteria involve only quantitative non-functional properties of web services, [20].

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 real time streaming web services.

We use the vector $Q = \{Q_1(s), \ldots, Q_R(s)\}$ to represent the QOS attributes of a service s, where the function $Q_i(s)$ 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 users feedbacks or social networks (e.g. reputation) [17].

The set of QOS attributes involves 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.

3.2 The aggregation functions

The web service composition has only four basic models: the sequential mode, the parallel mode, the selection mode, and the loop mode,

The QOS value of a composite service depends on the QOS values of its components as well as the composition model.

In this work, we consider only the sequential mode.

The Other models can be treated by using other Techniques [17].

The QOS vector for a composite service c is defined as $QOS'(c) = \{Q'_1(c), \ldots Q'_R(c)\}$. $Q'_i(c)$ 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.

These functions are shown in the following table[13].

Table 1. The aggregation functions

QOS Criterion	Aggregation function
Response Time	$Q'_1(C) = \sum_{j=1}^{n} Q_1(sj)$
Reputation	$Q'_2(C) = 1/n * \sum_{j=1}^{n} Q_2(sj)$
Price	$Q'_3(C) = \sum_{j=1}^{n} Q_3(sj)$
Reliability	$Q'_4(C) = \Pi_{j=1}^{n} Q_4(sj)$
Availability	$Q'_5(C) = \Pi_{j=1}^{n} Q_5(sj)$

3.3 The Global Constraints Modeling

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 in the section 3.1, we consider only positive QOS criteria. Therefore we have only lower bound constraints.

Let CONS= {cons1, ..consm…, consR }, $0 \leq m \leq R$, be a vector of global constraints (CONS is a vector of real values). Let c a concrete composition, in which a concrete web service is associated for each stage.

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

3.4 The Objective Function Modeling

Our objective function F1 contains several parts, each of them handles one aspect of the composition problem and must maximized.

$$F1(c, request) = w_1 * U_1(c) + w_2 * U_2(c, request) + w_3 * U_3(c, request) + w_4 * U_4(c)$$

The sum of the different weights is equal to 1.

In addition to that, the solution c must satisfy a set of global constraints:

$Q'_k(c) \geq Cons(k), \forall Cons(k) \in CONS$

The function U1(c) gives a real score, that represents the weighted sum of the different QOS values [24]. The score 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 the maximum aggregated values of the k-th QoS attribute of c are computed as follows:

$$Qmin'(k) = n * Qmin(k) \quad \ldots\ldots (1)$$

$$Qmax'(k) = n * Qmax(k)$$

$$Qmin(k) = Min(Q_k(s_i)) \quad / \forall s_i \in benchmark \quad \ldots\ldots (2)$$

$$Qmax(k) = Max(Q_k(s_i)) \quad / \forall s_i \in benchmark$$

Where Qmin(k) is the minimum value (e.g. minimum price) and Qmax(k) is the maximum value (e.g. maximum price) contained in the benchmark of services.

n denotes the size of the composition c.

The overall utility of a composite service c is computed as follows:

$$U_1(c) = \sum_{k=1}^{R} w'_k * \frac{\left(Q'_k(c) - Q\min'(k) \right)}{Q\max'(k) - Q\min'(k)} \quad ...(3)$$

with $w'_k \in R+$ and $\sum_{k=1}^{R} w'_k = 1$

w'_k are the weights (importance) of Q'_k.

$$U2(c,request) = \begin{cases} 1 \text{ if } inputs(firsts(c)) \equiv inputs(request) \\ 0.75 \text{ if } inputs(firsts(c)) \supseteq inputs(request) \\ -100 \text{ otherwise// we penalize a solution that violates the semantic compatibility with the request.} \end{cases}$$

$$U3(c,request) = \begin{cases} 1 \text{ if } outputs(lasts(c)) \equiv outputs(request) \\ 0.75 \text{ if } outputs(lasts(c)) \subseteq outputs(request) \\ -100 \text{ otherwise// we penalize a solution that violates the semantic compatibility with the request.} \end{cases}$$

$$U_4(c) = \frac{1}{n-1} * \sum_{i=1}^{n-1} closeness(OComp(c,i), IComp(c,i+1))$$

$$closeness(C1,C2) = \begin{cases} 1 \text{ if } C1 \equiv C2 \\ 0.75 \text{ if } C2 \supseteq C1 \\ -100 \text{ otherwise// we penalize a Solution that violates the semantic compatibility with the request.} \end{cases}$$

firsts(c) gives the atomic services of the first execution stage of the workflow c

lasts(c) gives the atomic services of the flast execution stage of the workflow c

OComp(c,k) denotes the outputs of the k^{th} component of c

IComp(c,k) denotes the inputs of the k^{th} component of c

If we have to compare the closeness between a set of concepts, instead of a couple of concepts, then we can use the algorithm presented in [3] in order to extend the functions U2, U3, Closeness.

A concrete composition c is optimal if it is feasible and if it has the maximum value of the function F1. We notice that the selection of the optimal concrete composition is NP hard (we must enumerate an exponential number of cases).

In case to hide the global constraints, we create a penalty function P, which promotes the feasible solutions and penalizes the non feasible solutions.

Roughly speaking, P(x) decreases the utility score 'F1(x,request)'of the solution that violates the global constraints. Several penalty functions are proposed in the literature [25,26], (we have static, dynamic, adaptive.. functions), for the sake of simplicity we have chosen a static function, because the two others does not give a significant improvement (they only increase the execution time):

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

Where

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

This formula means that a rigorous penalty is applied when we have a solution that violates a given constraint.

Finally the fitness employed by the BSO algorithm is defined as follows:

$$F2(c, request) = F1(c, request) + P(c)$$

4. THE PROPOSED APPROACH

BSO [10], is a social discrete algorithm, that uses the neighborhood to get a near optimal solution.

The population has a full mesh topology (each boid can communicate with entire swarm).

For the sake of simplicity, we suppose that the boid's position is a vector that contains 10 elements, each element denotes a service identifier, (or -1 if the service is absent), ie our composition contains less or equal than 10 elements. (Because our benchmark contains only 10 abstract classes).

The BSO version is given below:

1-Initialize the swarm, P(t), of boids such that the position xi (t) of each particle Pi ∈ P(t) is random within the hyperspace, with t = 0.

2. Evaluate the performance F2(Xi(t),request) of each boid, using its current position Xi(t).

3. Compare the performance of each individual to its best performance thus far:

if F2(Xi(t),request) > pbesti then

(a) pbesti = F2(Xi(t),request)

(b) Xpbesti = Xi(t)

4. Compare the performance of each boid to the global best boid (of the swarm):

if F2(Xi(t),request) > gbest then

(a) gbest = F2(Xi(t),request)

(b) Xgbest = Xi(t)

5. Compute the cohision of the swarm

$$coh = round(\frac{1}{size(P(t))} \sum_{i=1}^{size(P(t))} X_i(t))$$

6. Compute the alignment of the swarm

$$align = round(\frac{1}{size(P(t))} \sum_{i=1}^{size(P(t))} Xpbest_i(t))$$

7. Change the position for each boid using:

Select an abstract class h1 randomly from {1..10}

Select an abstract class h2 randomly from {1..10}

Select an abstract class h3 randomly from {1..10}

Select an abstract class h4 randomly from {1..10}

(a) Xi(t)[h1] = coh[h1]

(b) Xi(t)[h2] = align[h2]

(c) Xi(t)[h3] = xgbest(t)[h3]

(d) Xi(t)[h4] = random_instance_from_class(h4)

7. Move to the nest iteration: t = t + l

8. Go to step 2, and repeat until convergence or t=MaxIteration.

As mentioned above, we have 04 types of moving operators:

The first one (a) tends to be closer to the gravity center of the swarm.

The second one (b) tends to be closer to the average best position thus far (the gravity center of the different best positions (for all particles)).

The third one (c) tends to be closer to the best global position thus far (the best position of the entire swarm).

The fourth one (d) models a random moving or velocity (a sort of mutation).

5. THE EXPERIMENTATION

In case to test the performance of the BPO algorithm we use a benchmark inspired from the web service challenge[1].

in fact we use a subset which contains one request, 158 services, and an ontology that contains 500 concepts, the request accepts several solutions, the smallest solution contains at least 03 services (or 03 stages).

The space search is very huge, indeed we have 158!+157!....+1! candidate solutions.

The four sub-functions Ui have the same priority, ie w_i=0,25. In addition to that, the number of QOS attributes R is fixed to 5. All the QOS properties have the same priority (w'_k=0.2).

Several parameters have been modified to search the best results (in terms of optimality):

[1] http://www.ws-challenge.org/

1-The maximum number of iterations: it ∈ [100 , 10000],

2-The population size (the number of particles) ps ∈ [5 , 1000]

3-We suppose also that the optimality is defined as follows: rate=(the fitness of the current solution)/(the fitness of optimal solution).

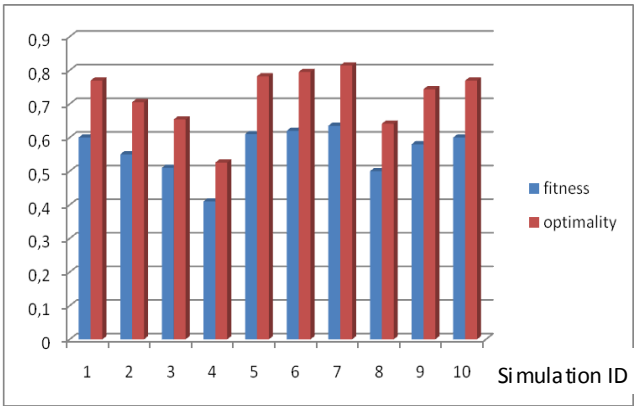


Fig 3: The optimality rates (with constraints, ps=1000)

Several simulations have been made with different configurations the results are resumed as follows:

The figure 3 shows 10 simulations, we notice that we can reach an optimality rate close to 81% .

Roughly speaking the BPO algorithm is slower than The SPO, but it is more immune to local optimums, than the second one. This ability is mainly due to the sophisticated moving operators (alignment, mutation and cohesion).

6. CONCLUSION

This paper presented a boid particle optimization approach for web service composition. The proposed algorithm shows a high ability for handling large spaces.

Our future research work will be focused on comparing the other techniques, mainly we will consider the constraint programming algorithms, the harmony search, and the stochastic optimization

We can apply also these algorithms in the multi-objective schema and compare the obtained results.

7. REFERENCES

[1] 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).

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

[3] F. Hadjila, Chikh A, A Belabed Semantic Web Service Composition: a Similarity Measure Based Approach Algorithm In Proceedings of ICIST'11 Tebessa Algeria 2011.

[4] D. Berardi, D. Calvanese, G. De Giacomo, R. Hull, andM.Mecella. Automatic composition of transition-

based semantic web services with messaging. In Proceedings of the 31st VLDB Conference. on Very Large Data Bases (VLDB 05), pages 613–624, Tronheim, Norway, 2005. ACM.

[5] Steffen Bleul, ThomasWeise, and Kurt Geihs. Making a fast semantic service composition system faster. In Proceedings of IEEE Joint Conference (CEC/EEE 2007) on E-Commerce Technology (9th CEC'07) and Enterprise Computing, E-Commerce and E-Services (4th EEE'07), 2007, pages 517–520. edings

[6] Weise T. Steffen B, Kurt G . Web Service Composition Systems for the Web Service Challenge – A Detailed Review. A technical report number: urn:nbn:de:hebis:34-2007111919638 university of kassel.

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

[8] F. Hadjila, Chikh A, M. Dali Yahiya QoS-aware Service Selection Based on Genetic Algorithm In Proceedings of CIIA'11 Saida Algeria 2011.

[9] F. Hadjila, Chikh A, M. Merzoug, Z Kameche QoS-aware Service Selection Based on swarm particle optimization In Proceedings of IEEE ICITES'12 Sousse Tunisia 2012

[10] Reynolds CW (1987) 'Flocks, Herds, and Schools: A Distributed Behavioral Model', Computer Graphics, vol.21, no.4, pp.25–34.

[11] J Kennedy, RC Eberhart, Particle Swarm Optimization, Proceedings of the IEEE International Conference on Neural Networks, Vol 4, pp 1942–1948, 1995.

[12] 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.

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

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

[15] 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.

[16] D. Ardagna and B. Pernici. Adaptive service composition in flexible processes. IEEE Transactions on Software Engineering, 33(6):369–384, 2007. Dustdar, S. and Schreiner, W. 'A survey on web services composition', Int. J. Web and Grid Services, Vol. 1, No. 1, pp.1–30. (2005).

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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. [74]. 2nd place in 2007 WSC. Online available at http://www.it-weise.de/documents/files/BWG2007WSC.

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

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

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

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

[26] 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.