Scalable Computing: Practice and Experience
Volume 12, Number 3, pp. 363–369. http://www.scpe.org
ISSN 1895-1767

A HYBRID FIREFLY-INSPIRED APPROACH FOR OPTIMAL SEMANTIC WEB SERVICE COMPOSITION

CRISTINA BIANCA POP, VIORICA ROZINA CHIFU, IOAN SALOMIE, RAMONA BIANCA BAICO, MIHAELA DINSOREANU, GEORGIANA COPIL *

Abstract. Inspired from biology, in this paper we propose a hybrid firefly method for selecting the optimal solution in semantic Web service composition. In our approach, the search space of the selection method is represented by an Enhanced Planning Graph structure which encodes all the Web service composition solutions for a given user request. As selection criteria we have considered the *QoS* attributes of the services involved in the composition as well as the semantic similarity between them. For the evaluation of the proposed selection method we have implemented an experimental prototype and carried out experiments on a scenario from the trip planning domain.

Key words: semantic Web service composition, firefly-based search, genetic operators, QoS, semantic similarity

AMS subject classifications.

1. Introduction and Related Work. The selection of the optimal solution in semantic Web service composition can be seen as an optimization problem which requires specific selection techniques that provide the desired results in an efficient way. Recent research studies demonstrated that principles inspired by the biological systems have lead to the design of efficient techniques that can be used to solve optimization problems. These biologically-inspired techniques are advantageous since they are capable of converging towards the optimal or a near-optimal solution in a short time without processing the entire search space. Such meta-heuristics include Genetic Algorithms, Ant Colony Optimization [3], Particle Swarm Optimization [5], or Artificial Immune Systems [1]. These meta-heuristics have been successfully applied to the problem of selecting the optimal solution in Web service composition.

In [8] genetic algorithms are used to find the optimal composition solution. The composition method is based on a given service abstract workflow, where each abstract service has a set of concrete services with different *QoS* values associated. Genetic algorithms are used to bind concrete services to the abstract ones aiming at identifying the optimal workflow instance in terms of *QoS* attributes. The genome is encoded as a binary string, where each position is associated to an abstract service in the workflow and indicates the concrete service which is selected to be used. The approach uses genetic operators and fitness functions applied on the genome to find the optimal composition solutions. The random mutation operator is used to generate new workflow instances. In the case of fitness functions, the approach presented in [8] proposes three fitness functions, one associated to each considered *QoS* attribute, in order to increase the probability of finding the optimal composition solution.

In [11] a method based on Ant Colony Optimization has been proposed for selecting the optimal service composition which uses the *QoS* attributes as selection criteria. The approach applies an ant-based selection method on a composition graph, where the graph nodes represent abstract services having associated sets of concrete services, while the edges denote the interactions between services. The *QoS* model defined in [11] makes distinction between the *QoS* attributes that should be minimized or maximized. In addition, the multi-pheromone concept is introduced for representing the *QoS* attributes.

A hybrid method combining Particle Swarm Optimization (PSO) [5] with Simmulated Annealing is proposed in [4] for selecting the optimal or a near-optimal service composition solution based on *QoS* attributes. Authors model service composition using an abstract workflow on which concrete services are mapped. A composition solution is considered as the position of a particle in PSO, while velocity is used to modify a composition solution. To avoid the problem of premature stagnation in a local optimal solution, a Simmulated Annealing-based strategy is introduced which produces new composition solutions by randomly perturbing an initial solution.

In [9], an immune-inspired selection algorithm is proposed in the context of Web service composition. The approach uses an abstract composition plan which is mapped to concrete services, obtaining in this way a graph structure having services in the nodes and *QoS* values on the arcs. The immune-inspired algorithm is applied

*Department of Computer Science, Technical University of Cluj-Napoca, 26-28 Baritiu str., Cluj-Napoca, Romania, ({Cristina.Pop, Viorica.Chifu, Ioan.Salomie}@cs.utcluj.ro).

to select the optimal composition solution based on QoS attributes. A composition solution is encoded using a binary string and a mutation operator is used to generate new composition solutions.

In this paper we present a hybrid method for selecting the optimal composition solution that combines an extended version [6] of the Firefly Search algorithm [10] with genetic operators. The search space for the hybrid method is represented by an Enhanced Planning Graph (EPG) which encodes all the service composition solutions for a given user request. In our approach, a user request is described in terms of functional and non-functional requirements. To identify the optimal solution encoded in the EPG, we define a fitness function which uses the QoS attributes and the semantic quality of the services involved in composition as selection criteria. The proposed selection method was tested on a scenario from the trip planning domain.

The paper is structured as follows. Section 2 presents the formal model for representing semantic Web service composition. Section 3 details the Hybrid Firefly method for selecting the optimal solution in Web service composition. Section 4 presents the performance evaluation of the proposed selection method. We end our paper with conclusions.

2. Semantic Web Service Composition Model. In our approach, Web service composition is modeled using an *Enhanced Planning Graph* (EPG) structure [7]. This graph actually represents the search space of the hybrid firefly selection method. The EPG is obtained by mapping the classical AI planning graph to the semantic Web service composition domain and by adding new domain related structures to ease the service composition representation and service selection. The construction of the EPG is an iterative process which is applied at the semantic level by considering the ontology concepts that annotate the services functionality and their input/output parameters. In each iteration, a new layer consisting of a tuple (A_i, L_i) is added to the graph where: (1) A_i contains clusters of services whose inputs are provided by the services from the previous layers and (2) L_i contains clusters of service parameters. A cluster of services groups services which provide similar functionality, while a cluster of service parameters groups similar input and output parameters. The first graph layer is represented by the tuple (A_0, L_0), where A_0 is an empty set of service clusters and L_0 contains the user-provided input parameters. The construction of the EPG ends either when the user requested outputs are contained in the current set of service parameters or when the graph reaches a fixed point. Reaching a fixed point means that the sets of clusters of services and parameters are the same for the last two consecutive generated layers.

A composition solution encoded in the EPG consists of a set of services, one from each cluster from each EPG layer.

3. The Hybrid Firefly Selection Method. The hybrid method for selecting the optimal solution in semantic Web service composition combines a firefly algorithm [6] with principles from evolutionary computing. We have proposed such a hybrid method to maintain a good balance between exploration and exploitation thus eliminating the problem of local optimum stagnation.

3.1. Problem Formalization. The firefly meta-heuristic relies on a set of artificial fireflies which communicate with each other to solve optimization problems. The behavior of artificial fireflies is modeled according to the behavior of fireflies in nature, which search for a mating partner by emitting a flashing light. In this section we present how we mapped the concepts of the firefly meta-heuristic to the problem of Web service composition.

Just as the real fireflies search for a mating partner by means of flashing lights, we have a number of artificial fireflies which search for the optimal service composition solution. Thus, we map the attraction behavior of fireflies to the problem of selecting the optimal service composition as follows: (i) a firefly becomes an artificial firefly, (ii) the position of a firefly becomes a service composition solution, (iii) the brightness of a firefly becomes the quality of a composition solution evaluated with a multi-criteria fitness function, (iv) the attractiveness between two fireflies becomes the similarity between two composition solutions, (v) the movement of a firefly is mapped to a modification of the firefly's current composition solution, (vi) the environment in which fireflies fly is mapped to the EPG.

We formally define an artificial firefly as follows:

$$firefly = (sol, score) \tag{3.1}$$

where sol is a service composition solution and $score$ is the quality of sol.

A service composition solution is defined as:

$$sol = \{solElem_1, ..., solElem_n\} \tag{3.2}$$

where (i) $solElem_i$ is a solution element composed of a set of services, one service from each cluster of layer i; and (ii) n is the total number of layers in the EPG.

To evaluate the score of a composition solution, we define a fitness function QF which considers the QoS attributes of the associated services as well as the semantic quality of the connections between these services:

$$QF(sol) = \frac{w_{QoS} * QoS(sol) + w_{Sem} * Sem(sol)}{(w_{QoS} + w_{Sem}) * |sol|} \tag{3.3}$$

where: (i) $QoS(sol)$ citepop2010 is the QoS score of the composition solution sol; (ii) $Sem(sol)$ [7] is the semantic quality score of the solution sol; (iii) w_{QoS} and w_{Sem} are the weights corresponding to user preference related to the relevance of QoS and semantic quality.

3.2. The Hybrid Firefly Selection Algorithm. A prerequisite of the hybrid selection method is to establish the number of fireflies that will be used in the search process so as to obtain the optimal solution in a short time interval and without processing the entire search space. We have defined the number of fireflies based on the total number of solutions encoded in the EPG:

$$noF = Round(\sqrt[n]{noSol}) \tag{3.4}$$

where: (i) $noSol$ is the number of possible composition solutions encoded in the EPG; and (ii) n is a positive integer determined experimentally.

The inputs of the selection algorithm (see *Algorithm_1*) are: (i) the EPG resulted from the Web service composition process, (ii) the weights w_{QoS} and w_{Sem} which state the relevance of a solution's QoS quality compared to its semantic quality, and (iii) a number noF (formula 3.4) of artificial fireflies used to search for the best composition. The algorithm returns the optimal or a near-optimal composition solution.

Algorithm 3.2.1 Hybrid_Firefly_Web_Service_Selection

```
1  Input: EPG, w_QoS, w_Sem, noF
2  Output: fSol_best
3  begin
4  FSOL = ∅
5  for i = 1 to noF do FSOL = FSOL ∪ Gen_Random_Solution(EPG)
6  repeat
7      for i = 1 to noF do
8          for j = 1 to noF do
9              if (QF(FSOL[i]) < QF(FSOL[j])) then
10                 r = Compute_Distance(FSOL[i], FSOL[j])
11                 FSOL[i] = Crossover(FSOL[i], FSOL[j], r)
12                 u = Generate_Random_Vector(|FSOL[i]|)
13                 FSOL[i] = Mutation(FSOL[i], u)
14             end if
15         end for
16     end for
17     fSol_best = Get_Best_Solution(FSOL)
18     SOL_best = SOL_best ∪ fSol_best
19     u = Generate_Random_Vector(|fSol_best|)
20     FSOL = Modify_Best_Firefly(FSOL, u)
21 until (Stopping_Condition())
22 return Get_Best_Solution(SOL_best)
23 end
```

In the first step of the selection algorithm each firefly is associated with a randomly generated composition solution (see line 5). These initial solutions are further improved in an iterative process which stops when the best solution has been the same over the last $noIt$ iterations (see line 21).

In each iteration, if the score of the solution associated to a firefly is better than the score of the solution associated to another firefly it means that the latter firefly will be attracted towards the first one and thus it will have its solution improved. The steps for improving the solution associated to the less bright firefly are the following:

1. The distance r between the two composition solutions is computed (see line 10) as the difference of their scores.
2. A crossover operator is applied between the two composition solutions in a number of points depending on the value of the distance r compared to three thresholds r_1, r_2, r_3 (see line 11). As a result of the crossover operation, two new solutions will be obtained and the one having the highest score according to the QF function will be kept.
3. A mutation operation (see line 13) is performed on the best solution obtained within crossover to introduce diversity. In the mutation process, a mutation vector is randomly generated (see line 12) to specify the points where services will be replaced with other services from the same clusters.

After all solutions have been processed, the best one is determined (see line 17), added to the set of the best composition solutions and then mutated (see line 20) according to a randomly generated mutation vector (see line 19). This last mutation is performed to enlarge the search space and to avoid the stagnation in a local optimal solution.

4. Performance Evaluation. To validate our selection approach we have implemented an experimental framework which has been tested on a set of scenarios from the trip planning domain. First we performed a series of experiments aiming to identify the optimal values of the selection method's adjustable parameters. Then, using the optimal configuration of the adjustable parameters we performed further tests to evaluate the performance of the proposed method in contrast with a Bee-inspired selection method.

4.1. Experimental Framework. The architecture of the experimental framework is presented in Figure 4.1.

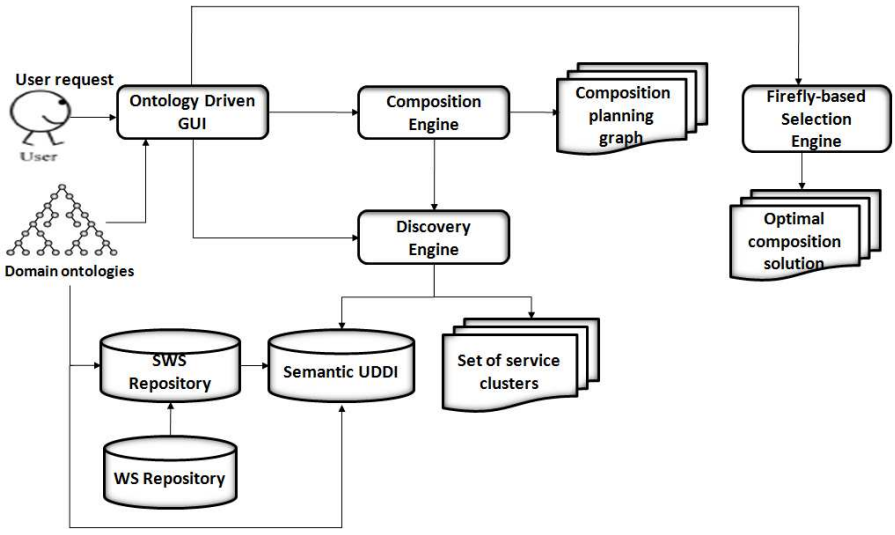

Fig. 4.1: The architecture of the experimental framework.

The ontology driven graphical user interface guides the users in the processes of searching and composing Web services by providing a controlled language that uses the ontology concepts. The SWS Repository is a repository of semantically annotated services based on the Domain Ontology. The Semantic UDDI extends the classical UDDI structure by (*i*) storing semantic Web service descriptions and (*ii*) providing means to semantically inquire the UDDI repository. The Discovery Engine receives a service request from a user or from the Composition Engine. In the case of a user request, the Discovery Engine provides an ordered set of atomic Web services which match the request. In the case of a service request from the Composition Engine, the Discovery Engine provides a set of atomic Web services organized in clusters. To satisfy the service requests,

the Discovery Engine inquires the Semantic UDDI. The Composition Engine is the component responsible for the construction of the EPG. The Composition Engine interacts with the Discovery Engine which provides the appropriate service clusters. The EPG is then used by the Firefly-based selection engine which provides the optimal or a near optimal composition solution according to the user preferences.

4.2. Setting the Optimal Values of the Adjustable Parameters. The framework has been tested on a scenario (see in Table 4.1 the associated user request) from the trip planning domain for which the EPG is organized on 3 layers consisting of 51 services grouped in 11 clusters. This EPG encodes 13996800 composition solutions.

Table 4.1: User request for planning a holiday

User inputs	Requested outputs	*QoS* weights	Semantic quality weight
SourceCity, DestinationCity, StartDate, EndDate, HotelType, NumberOfPersons, NumberOfRooms, CarType, ActivityType	AccomodationInvoice, FlightInvoice, CarInvoice	Total $QoS = 0.55$, Availability = 0.30, Reliability = 0.30, Cost = 0.15, Response time = 0.25	0.35

To identify the optimal values of the selection method's adjustable parameters (the number of stagnations, noS, the three thresholds, r_1, r_2 and r_3, for computing the number of crossover points) we have performed 100 runs of the associated selection algorithm for each configuration and analyzed how the average number of processed solutions ($noSolGen_{avg}$), the average simulation time ($tExec_{avg}$), and the standard deviation ($stDev_{avg}$) are affected. Table 4.2 presents the average values obtained for each configuration considered.

Table 4.2: Tests summary for the Hybrid Firefly Algorithm

#	noS	r_1	r_2	r_3	$noSolGen_{avg}$	$tExec_{avg}$	$stDev_{avg}$
1	3	0.001	0.005	0.01	2103	8	0.0084
2	3	0.003	0.006	0.008	2186	9	0.0089
3	3	0.03	0.06	0.09	2146	8	0.0126
4	3	0.01	0.05	0.099	2192	8	0.0111
5	6	0.001	0.005	0.01	2145	9	0.0076
6	6	0.003	0.006	0.008	2120	9	0.0089
7	6	0.03	0.06	0.09	2119	8	0.0089
8	6	0.01	0.05	0.099	2098	7	0.0111
9	9	0.001	0.005	0.01	2079	8	0.0098
10	9	0.003	0.006	0.008	2079	8	0.0107
11	9	0.03	0.06	0.09	2070	8	0.0125
12	9	0.01	0.05	0.099	2140	8	0.0108

When choosing the final values of the adjustable parameters a tradeoff between obtaining the optimal solution and a very low execution time has to be considered. By analyzing the results from Table 4.2 it can be noticed that the smallest average execution time, 7 seconds, corresponds to a standard deviation of the

identified optimal solution of 0.0111, but for a standard deviation of 0.0076, which is the smallest one, the average execution can get up to 9 seconds.

By analyzing the experimental results we conclude that the Hybrid Firefly selection algorithm returns the optimal or a near-optimal solution (the average standard deviation is 0.089) on average in 8 seconds by processing around 0.015% of the search space.

4.3. Comparative Analysis. To assess the performance of the Hybrid Firefly selection algorithm we have compared it with a Bee-inspired selection algorithm we previously introduced in [2]. The two selection algorithms have been comparatively evaluated according to the following criteria: the average number of processed solutions, the average simulation time, and the standard deviation of the score of the best solution returned by the algorithm related to the score of the global optimal composition solution for the considered scenario.

For each algorithm we performed the same number of simulations on the same scenario and on the same machine. In Table 4.3 we summarize the experimental results obtained for the two selection algorithms taking into consideration the optimal values of their adjustable parameters.

By analyzing the experimental results we conclude that the Hybrid Firefly algorithm outperforms the Bee-inspired algorithm in terms of number of processed solutions and executon time, but in terms of standard deviation it provides slightly higher values which are acceptable.

Table 4.3: Comparative analysis between the Hybrid Firefly Selection Algorithm and the Bee-inspired Selection Algorithm

Method	Average number of processed solutions	Average execution time	Average standard deviation
Hybrid Firefly	2145	9	0.007
Bee-inspired	2867	13	0.002

5. Conclusions and Future Work. This paper proposed a firefly-inspired method for selecting the optimal or a near optimal solution in semantic Web service composition. The selection method has been applied on Enhanced Planning Graph which encodes the set of composition solutions for a given user request. By combining the firefly-based selection approach with genetic operators we ensure a good balance between exploration and exploitation thus avoiding the problem of stagnation in a local optimum. To demonstrate the feasibility of our approach, we have implemented an experimental prototype which has been tested on a scenario from the trip planning domain. To assess the performance of the Hybrid Firefly method we have compared it with a Bee-inspired selection method.

REFERENCES

[1] L. N. Castro and F. von Zuben, *Learning and Optimization using the Clonal Selection Principle*, IEEE Transactions on Evolutionary Computation, Volume 6, Issue 3, pp. 239-251, 2002.

[2] V. R. Chifu, C. B. Pop, I. Salomie, M. Dinsoreanu, A. N. Niculici and D. S. Suia, *Selecting the Optimal Web Service Composition based on a Multi-criteria Bee-inspired Method*, Proceedings of the 12th International Conference on Information Integration and Web-based Applications & Services, pp. 40-47, 2010.

[3] M. Dorigo, M. Birattari, and T. Stutzle, *Ant Colony Optimization - Artificial Ants as a Computational Intelligence Technique*, IEEE Computational Intelligence Magazine, 1(4), pp. 28-39, 2006.

[4] X. Fan and X. Fang, *On Optimal Decision for QoS-Aware Composite Service Selection*, Information Technology Journal, Volume 9, Issue 6, pp. 1207-1211, 2010.

[5] J. Kennedy, and R. Eberhart, *Particle swarm optimization*, Proceedings of the IEEE International Conference on Neural Networks, pp. 1942-1948, 1995.

[6] S. Lukasik and S. Zak, *Firefly Algorithm for Continuous Constrained Optimization Tasks*, Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems, LNCS vol. 5796, pp. 97-106, 2009.

[7] C. B. Pop,V. R. Chifu, I. Salomie, and M. Dinsoreanu, *Immune-Inspired Method for Selecting the Optimal Solution in Web Service Composition*, LNCS, vol. 6162, pp. 1-17, 2010.

[8] J. Wang and Y. Hou, *Optimal Web Service Selection based on Multi-Objective Genetic Algorithm*, Proceedings of the International Symposium on Computational Intelligence and Design - Volume 01, pp. 553-556, 2008.

[9] J. Xu and S. Reiff-Marganiec, *Towards Heuristic Web Services Composition Using Immune Algorithm*, Proceedings of the International Conference on Web Services, pp. 238-245, 2008.

[10] X.S. Yang, *Nature-Inspired Metaheuristic Algorithms*, Luniver Press, 2008.
[11] W. Zhang, C. K. Chang, T. Feng and H. Jiang, *QoS-based Dynamic Web Service Composition with Ant Colony Optimization*, Proceedings of the 34th Annual IEEE Computer Software and Applications Conference, pp. 493-502, 2010.

Edited by: Dana Petcu and Jose Luis Vazquez-Poletti
Received: August 1, 2011
Accepted: August 31, 2011