http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 10 (1): 113-119, 2011
ISSN 1812-5638

Multi-path QoS-Aware Web Service Composition using Variable Length Chromosome Genetic Algorithm

Honghong Jiang, Xiaohu Yang, Keting Yin, Shuai Zhang and Jerry A. Cristoforo
Room 189, Dorm 8, Zhejiang University, Hangzhou, Zhejiang, 310027, China

Abstract: In this study, we proposed to adopt a variable length chromosome Genetic Algorithm (GA) for handling QoS-aware service composition among multiple paths (multi-path) problem. Our approach uses variable length chromosomes to represent composited services in multiple paths and conducts the gene crossover operation based on service parameters matching. The scalability of the algorithm is analyzed theoretically and its effectiveness is demonstrated by experimental results.

Key words: Service composition, QoS-aware, genetic algorithm, multi-path, VLCGA, scalability

INTRODUCTION

Heterogeneous services over internet are provided to implement various functions. A collection of services which have identical functionality can be abstracted as an abstract service. And in Service Oriented Architectures (SOA) several abstract services can be integrated to perform complex business transaction which cannot be implemented by one single abstract service.

Web services integration approaches can generate many composite web services for the same functional requirement. These compositions may have different composite structures of abstract services (paths). Plenty of web services with different QoS properties are available for a certain abstract service.

Our research mainly addresses following key issues for QoS-aware web service composition:

- **Multi-path QoS-aware service composition:** As mentioned previously, lots of composite services can implement the same required functionality. Composite web services are discriminated by their QoS which are non-functional properties such as price, response time, availability and reputation (O'Sullivan *et al*., 2004). The overall QoS of composition relies on the QoS of individual services in composition. And it should be defined from user's perspective so that user can define the QoS requirements. Furthermore, QoS has multiple dimensions. All the dimensions should be integrated according to user's preference

Picking out concrete service composition plan which has optimum QoS and fulfils all the requested QoS constraints is the goal of QoS-aware service composition approach. For multi-path composition, all the paths should be included in the search space.

- **Flexible service composition:** Web is an open environment which evolves quickly, so does the web service over it. QoS of web service changes frequently. For example, service upgrade, network latency and web site failure can all lead to the service QoS changing. What's more, new services keep flowing into internet and old services keep flowing out. Thus, services need to be re-composed from time to time, which should take not only the changed service QoS but also the new/old service information into consideration

Quite a lot of approaches (Zeng *et al*., 2004; Ardagna and Pernici, 2007; Berbner *et al*., 2006; Canfora *et al*., 2005a; Yin *et al*., 2010; Yan-Ping and Zeng-Zhi, 2007) have been proposed for resolving QoS-aware web service composition problem. Recently, Integer Programming (IP) has been largely researched in literatures (Zeng *et al*., 2004; Ardagna and Pernici, 2007) for resolving this problem. Zeng *et al*. (2004) use the logarithmic reductions to aggregate the QoS value and linearize the constraints and objective to according with IP approach definition. Critical path for And/Or workflow is adopted in this model which affects the scalability and precision of model. Ardagna and Pernici (2007) propose a Mixed Integer Linear Programming (MILP) approach which involves multiple execution plans in IP approach for Or structure.

Canfora *et al*. (2005a) discuss GA as an approach for solving the QoS-aware Web Service composition problem.

Corresponding Author: Xiaohu Yang, Room 189, Dorm 8, Zhejiang University, Hangzhou, Zhejiang, 310027, China
Tel: +86-13588196867 Fax: +86-0571 28111775

GA shows a better performance and scalability than linear IP in Canfora's test results when increasing numbers of candidate Web Services and tasks. In his another study, the author considers re-composition at runtime as well (Canfora *et al*., 2005b).

However, most of the approaches discussed above are based on the single path search without taking multi-path into account. Although Zhang *et al*. (2006) have proposed a two-dimensional GA approach which uses relation matrix to resolve the multi-path web service composition problem, this approach cannot support certain complicate structures such as parallel branch in multi-path branch. And Dong (2009) transforms the multi-path service composition problem to shortest path in graph problem for resolving it, but he only considers the sequential composited structure.

In this study, a variable length chromosome GA (VLCGA) which can support multi-path service composition is proposed. This approach uses variable length chromosome to represent the concrete service composition plans in different paths. And the gene crossover operation is implemented based on syntactic matching. The outstanding features of VLCGA are:

VLCGA can support both single-path and multi-path QoS-aware service composition. In addition, VLCGA is compatible with complicated structures. It is independent of path structure.

VLCGA has good scalability for abstract service construction changing. Path modifying, deleting and new path adding can be all managed in the initialization stage conveniently. If services need to be re-composited, the remainder of services can be organized as a new multi-path service composition problem which can also be resolved by VLCGA.

It should be noted that this approach has been used in routing application (Xu and Xu, 2008). It is possible to leverage this approach to other distributed components QoS-aware composition problem.

Nevertheless, VLCGA has weaknesses as the GA has its own flaw for web service selection and composition, which will be discussed later.

MULTI-PATH QOS-AWARE SERVICE COMPOSITION

In syntactic matching mode, abstract services are orchestrated by matching their parameters (Beek *et al*., 2007; Kwon *et al*., 2008; Hewett *et al*., 2009) with each other. Several paths which satisfy the parameters requested by user (input parameters and output parameters) will be generated for one transaction.

Taking trip service as an example, given parameters are (Travel time, Departure point, Destination city) and requested parameters are (Airdrome, Flight number, Flight depart/land time, Hotel Price, Hotel Address, Shuttle bus station, Shuttle bus contact). The parameters of abstract services over internet are shown in Table 1.

Table 1: Abstract web service parameters

Abstract service	Input parameter	Output parameter
1	(Travel time, Departure point, Destination city)	(Air company) (Flight number)
2	(Air company)	
3	(Travel time, Departure point, Destination city)	(Air company, Flight number)
4	(Air company, Flight number)	(Airdrome, Flight depart/land time)
5	(Flight depart/land time)	(Hotel Name, Arrive/Leave date)
6	(Hotel Name, Arrive/Leave date)	(Hotel Price)
7	(Hotel Name)	(Hotel Address)
8	(Hotel Address, Arrive/Leave date)	(Shuttle bus station, Shuttle bus contact)
9	(Flight depart/land time)	(Hotel Availability, Arrive/Leave date)
10	(Hotel availability, Arrive/Leave date)	(Hotel name, Hotel Price, Hotel Address)
11	(Hotel Name)	(Shuttle bus number)
12	(Hotel Address)	(Shuttle bus number)
13	(Shuttle bus number, Flight depart/land time)	(Shuttle bus station, Shuttle bus contact)

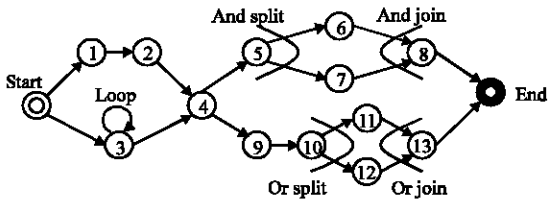

Fig. 1: Process flow for one customer requirement

After parameter matching, four paths are built for implementing user's requirement: {1, 2, 4, 5, 6, 7, 8}, {1, 2, 4, 9, 10, 11, 12, 13}, {3, 4, 5, 6, 7, 8} and {3, 4, 9, 10, 11, 12, 13}.

In our modeling, path is constituted of workflow patterns that represent basic structural elements. This concept is inspired by Van der Aalst *et al*. (2003). Four workflow patterns: Sequence and, Or and Loop will be researched in this study.

Using the same travel example, the process flow of travel service represented by state chart is shown in Fig. 1. Paths started from Start and end with End states which are labeled by hollow and solid concentric circles respectively. Circles represent abstract services and the arrows indicate the execution order. Curved arrow represents the Loop workflow pattern. And the branches between split and join sign consist And/Or workflow pattern. Other branches are the multi-path branches.

In Fig. 1, four paths: {Sequence [1, 2, 4, 5] and [6, 7], Sequence [8]}, {Sequence [1, 2, 4, 9, 10], Or [11, 12],

Table 2: Aggregation functions for each workflow construct and QoS attribute

QoS	Sequence	And	Or	Loop
P	$\sum_{i=1}^{n} q^{P}_{i}$	$\sum_{i=1}^{n} q^{P}_{i}$	$\sum_{i=1}^{n} p_i q^{P}_{i}$	$k*q^{P}_{i}$
T	$\sum_{i=1}^{n} q^{T}_{i}$	$\max\{q^{T}_{1}, \cdots q^{T}_{n}\}$	$\sum_{i=1}^{n} p_i q^{T}_{i}$	$k*q^{T}_{i}$
A	$\prod_{i=1}^{n} q^{A}_{i}$	$\prod_{i=1}^{n} q^{A}_{i}$	$\sum_{i=1}^{n} p_i q^{A}_{i}$	$(q^{A}_{i})^{k}$
R	$\frac{\sum_{i=1}^{n} q^{R}_{i}}{n}$	$\frac{\sum_{i=1}^{n} q^{R}_{i}}{n}$	$\sum_{i=1}^{n} p_i q^{R}_{i}$	q^{R}_{i}

Sequence [13]}, {Loop [3], Sequence [4, 5] and [6, 7], Sequence [8]} and {Loop [3], Sequence [4, 9, 10], Or [11, 12], Sequence [13]} started from the initial state and ended with the goal state can satisfy the desired functional requirement.

On the other hand, each abstract service in process flow has many service candidates with different QoS value. Picking one path and selecting one candidate for every abstract service in selected path will form a set of web service such as {Sequence [c1, c2, c4, c5] and [c6, c7], Sequence [c8]} which is so called composition plan. Composition plans have identical functionality but distinct QoS. QoS-aware service composition is aimed to find a composition plan which has optimum QoS on the premise of non-functional constraints obedience.

QoS aggregation for composite web service: The QoS of the composition web service is calculated by the QoS of the atomic services. Four quality dimensions are considered in this study: price (P), response time (T), availability (A) and reputation (R) which are frequently researched in other literatures (Zeng *et al*., 2004; Ardagna and Pernici, 2007; Alrifai and Risse, 2009). Table 2 shows the aggregation function of the four QoS attributes in different workflow patterns.

In Table 2, n is the number of services in one path, q_i represents the QoS value of web service i, the execution probability of web service i in Or workflow pattern is p_i and k is the loop times for service i.

VARIABLE LENGTH CHROMOSOME GA DISCUSSION

GA is a search technique used to find exact or approximate solutions to optimization and search problems, in which a population of abstract representations of candidate solutions evolves toward better solutions (Ericross and Peter, 2002).

VLCGA is proposed in this study to resolve multi-path web service composition problem. Variable length chromosomes are used to represent composition plans in multiple paths. And the cut and splice operations for gene crossover are implemented by

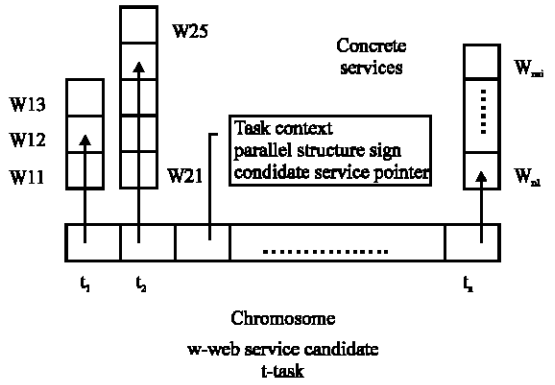

Fig. 2: Genetic representation

abstract service parameters matching. For brevity, abstract web service is referred as task in the remainder of study.

Representation: As described in the introduction, paths contain different number of tasks. Accordingly, composition plans have different length which will be denoted by variable length chromosome (Brie and Morignot, 2005).

As shown in Fig. 2, unit in chromosome corresponds to the selected candidate for one task. Content of unit is a triple: (Task Context, Workflow Sign, Candidate Service Pointer). In which, task context describes the input and output parameters of task, workflow sign is used to check whether this task is inside And/Or workflow and candidate service pointer is a pointer of the selected service for the corresponding task in composition plan.

Initialization: Initial chromosomes are built for different paths randomly. In order to cover all the paths, the initialization of chromosomes should contain all the path branches. The probability that chromosomes are built for each branch is calculated by Eq. 1:

$$\text{Chromosome build rate}_i = \frac{\sum_{j=1}^{J_i} n_{ij}}{\sum_{i=1}^{I}\sum_{j=1}^{J_i} n_{ij}} \quad (1)$$

where, J_i is the task number of branch i, n_{ij} is the candidate service number of task j in branch i and I is the total branch number.

According to Eq. 1, the initial chromosomes number on certain branch increases while the candidate web service number of this branch increases. This initial calculation is intended to average the selected probability of web services in each path branches during evolution.

Selection: The goal of web service composition is optimizing the composite plan's QoS without constraints violation. A fitness function is defined to indicate the fineness of composition plan. And fitness value is used to select the parent chromosomes for the next generation. So, the aggregated QoS value in four QoS dimensions should be all involved in fitness function. Besides, the QoS dispersion between the actual QoS and the QoS constraints should also be a fitness factor.

The fitness function is defined as Eq. 2 and 3:

$$\text{fitness}(cp) = \frac{\omega_1 Q^A(cp) + \omega_2 Q^R(cp)}{\omega_3 Q^P(cp) + \omega_4 Q^T(cp)} - \omega_5 D(cp) \quad (2)$$

$$D(cp) = \sum_{m=1}^{M'} \left(\frac{\Delta Q^m}{Q^m_{constraint}}\right)^2 \quad (3)$$

where, $Q^m_{(cp)}$ and $Q^m_{constraint}$ are the mth QoS value of composition plan and requested QoS constraint, ω is the weight of each QoS dimension which is defined by the users, M' is the number of QoS constraints, M' is the synthetical QoS dispersion which is calculated by ΔQ^m– the mth QoS dispersion that defined in Eq. 4 and 5.

Two kinds of QoS properties are researched: positive property such as availability which is better with higher value and negative property such as price which is better with lower value. QoS dispersion of two kinds QoS are calculated in two ways as following.

For negative properties:

$$\Delta Q^m = \begin{cases} Q^m(cp) - Q^m_{constraint}, & Q^m(cp) > Q^m_{constraint} \\ 0, & Q^m(cp) < Q^m_{constraint} \end{cases} \quad (4)$$

For positive properties:

$$\Delta Q^m = \begin{cases} Q^m_{constraint} - Q^m(cp), & Q^m(cp) < Q^m_{constraint} \\ 0, & Q^m(cp) > Q^m_{constraint} \end{cases} \quad (5)$$

where, $Q^m_{constraint}$ represent the mth QoS constraint which should also defined by users (e.g., 10 sec as the constraint for response time).

It is notable that the QoS dispersion function in Eq. 3 is static for every generation which may lead to the risk of discarding a composition plan which is actually close to an idea solution at early stage. To avoid this, a dynamic dispersion function is adopted to weaken the infection of dispersion function at early generations as shown in Eq. 6:

$$D(cp) = \frac{gen}{maxgen} \sum_{m=1}^{M'} \left(\frac{\Delta Q^m}{Q^m_{constraint}}\right)^2 \quad (6)$$

Table 3: Matching process pseudo code

```
Procedure Matching
/*Input: Input_ak = [p_1, p_2, ..., p_n] a set of input parameters of service_ak.*/
/*Output: An index for service_br.*/
Begin
    For i from 1 to n_b
      If w_bi is in And/Or workflow
/*Choose the service not in And/Or workflow.*/
        Continue;
      Else
        Inputi = {p| p∈Input_bi};
      End If
/*Get the input parameters of w_bi.*/
      For j from 1 to n
        match the p_j in Input_ak;
        If matched
          delete p_j in Input_i;
        Else
/*Make sure all the input parameters in
a are matched*/
          Throw un-matched error;
        End If
      End for
      If Input_i is null
/*Make sure all the input parameters in
b are matched*/
        Return i;
      Else
        Throw un-matched error;
      End If
  End For
End
```

where, gen is the current generation, while maxgen is the maximum generations.

Cut and splice: Cut and splice operations implement the crossover operation in tradition GA. Suppose the parents chromosomes are a and b. Firstly, the cut operation picks up one service randomly in a e.g., wak which is outside the And/Or workflow. Secondly, the input parameters of w_{ak} are matched with the contexts of services in b and the matched service is picked out e.g., wbr which is also not in And/Or workflow. The pseudo code of the matching process is as shown in Table 3:

Accordingly, the cut points of a and b are t_{ak} and t_{br}. The remainder genes from the cut points are exchanged between a and b as shown in Fig. 3 to produce two children chromosomes whose length may also be different.

Take the travel service as an example, two composition plans a: {ca1, ca2, ca4, ca5, ca6, ca7, ca8} on path {Sequence [1, 2, 4, 5] and [6, 7], Sequence [8]} and b: {cb3, cb4, cb5, cb6, cb7, cb8} on path {Loop [3], Sequence [4, 5] and [6, 7], Sequence [8]} are chosen for the parent chromosomes. In a, service ca4 is picked up as the cut point. The input parameters of ca4 are (Air company, Flight number) which can be matched by cb4's input parameters. So, the cut point of b should be cb4. After the splicing, two new children

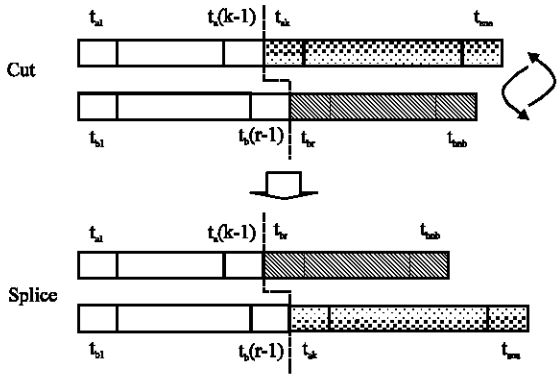

Fig. 3: Cut and splice operations

chromosome {ca1, ca2, cb4, cb5, cb6, cb7, cb8} and {cb3, ca4, ca5, ca6, ca7, ca8} are produced.

Mutation: Like the other GAs (Canfora *et al*., 2005a; Zhang *et al*., 2006), mutation operation randomly chooses a composition plan at specified rate and picks one concrete service which has more than one alternative in these. And then the concrete service is replaced by another random alternate candidate to finish the mutation.

SCALABILITY OF VLCGA

Algorithm can be well extended for path modifying, deleting and new path adding. These three kinds of path changing can all be coped with at the initialization stage in VLCGA. If some paths have been modified, algorithm can build the initial chromosomes on the modified paths. Adding the initial chromosomes at new paths can handle the path adding change, while deleting paths leads to the initial chromosomes' deleting at those paths. These operations can all be easily implemented by VLCGA.

Re-composition of services needs to be executed from time to time. It is also convenient for VLCGA to re-composite services. Remainder services in process flow after the last executed service can be organized to a new multi-path process flow. Using VLCGA for the new multi-path service composition is also feasible. Still take travel service as an example, suppose composition plan {Sequence [c1, c2, c4, c5] and [c6, c7], Sequence [c8]} is selected for execution and c6 becomes unavailable after service c4 finishes. This leads to a re-composition of services which are implemented after c4. The VLCGA orchestration for services in remainder process flow as shown in Fig. 4 can accomplish the re-composition job.

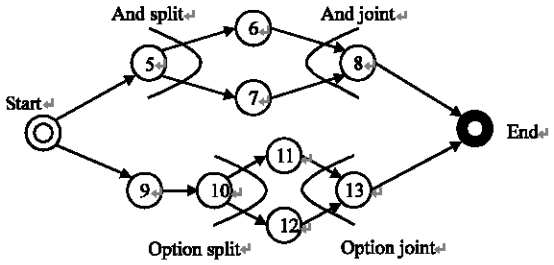

Fig. 4: Re-composition process flow

EMPIRICAL STUDY

A series of tests were carried out for comparing the resolve rate and efficiency between tradition GA and the VLCGA. For brevity, the And/Or workflows are considered as small composite web services and loop structure is unfolded as a sequence structure in experiments (Zeng *et al*., 2004).

All the experiments were conducted on a personal computer with 2 Intel Duo 2.33 GHz processors and 2 GB RAM. Tests were performed under Windows XP SP3, using Java 2 Standard Edition V1.6.2.

Experiment case: Two test cases were designed. One is for comparing the resolve rate between two algorithms, while the other aims at evaluating the efficiency of both algorithms. For tradition GA, one path is selected randomly for testing. And for VLCGA all the paths are involved in calculation.

Experiment result: For GA is a stochastic algorithm, it can not always find the solution in a fixed time. In the first experiment, we run the algorithms in a fixed time for 100 times and recorded the times that they find a solution as the resolve rate. In this experiment, task number is fixed to 40, candidate service number for each task is fixed to 40 and the calculate time is fixed to 1500 milliseconds.

Process flow has 12 paths in total. Constraint relaxation degree for time, availability and reputation are all set to 0.5. Only the constraint relaxation degree of price varies from 0.1 to 0.5.

Constraint relaxation degree represents the looseness of QoS constraints. For positive/negative QoS attribute, higher/lower constraint value means lower relaxation degree. The constraint relaxation degree for each QoS attribute is defined in Table 4.

where $q_{Max}^{x}{}_{i}$ is the max value of QoS x of candidate services for task i and $q_{Max}^{x}{}_{i}$ is the min value of QoS x of candidate services for task i.

Two algorithms run 100 times under the same QoS constraint in fixed time. The resolve rate indicates how many times algorithms find a solution in these 100 runs.

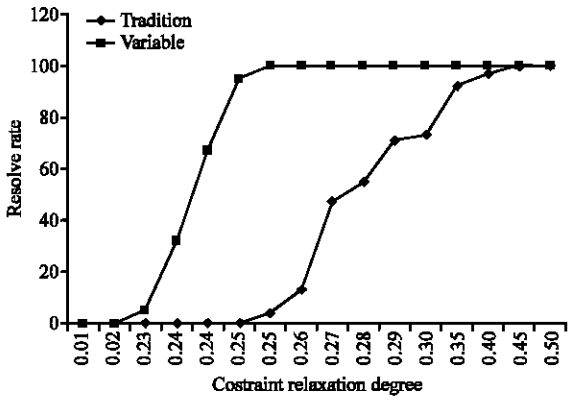

Fig. 5: Resolve rates of two algorithms

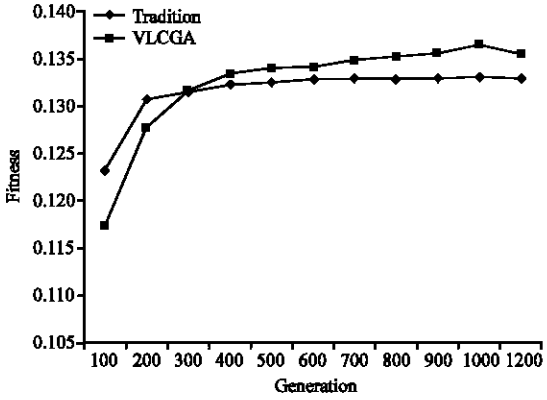

Fig. 6: Efficiency of two algorithms

Table 4: Relaxation degree for each QoS attribute

QoS	Relaxation degree
P	$\frac{\text{Constraint value}-\sum_{i=1}^{n} q_{\text{Min}}{}^{P}{}_{i}}{\sum_{i=1}^{n} q_{\text{Max}}{}^{P}{}_{i}-\sum_{i=1}^{n} q_{\text{Min}}{}^{P}{}_{i}}$
T	$\frac{\text{Constraint value}-\sum_{i=1}^{n} q_{\text{Min}}{}^{T}{}_{i}}{\sum_{i=1}^{n} q_{\text{Max}}{}^{T}{}_{i}-\sum_{i=1}^{n} q_{\text{Min}}{}^{T}{}_{i}}$
A	$\log_{\left(\frac{\prod_{i=1}^{n} q_{\text{Max}}{}^{A}{}_{i}}{\prod_{i=1}^{n} q_{\text{Man}}{}^{A}{}_{i}}\right)}\left(\frac{\text{Constraint value}}{\prod_{i=1}^{n} q_{\text{Min}}{}^{A}{}_{i}}\right)$
R	$\frac{\text{n*Constraint value}-\sum_{i=1}^{n} q_{\text{Min}}{}^{R}{}_{i}}{\sum_{i=1}^{n} q_{\text{Mix}}{}^{R}{}_{i}-\sum_{i=1}^{n} q_{\text{Min}}{}^{R}{}_{i}}$

The rates of finding a resolution of two algorithms under each constraint were recorded. As shown in Fig. 5, VLCGA has higher resolve rate than tradition GA under same constraint in a fixed time. VLCGA starts to have solutions from relaxation degree 0.23 and achieves 100% resolve rate from relaxation degree 0.25. On the other hand, tradition GA starts to have solutions from relaxation degree 0.245 and achieves 100% resolve rate from relaxation degree 0.45 which is much higher than VLCGA. The test results show that the VLCGA can resolve the service composition problem under less relaxed constraint within a given period.

And in the other experiment for comparing two algorithms' efficiency, the task number is fixed to 10, candidate service number of each task is fixed to 10 and constraint relaxation degree for each QoS dimension are set to 0.5.

The path number in process flow is also 12. The fitness values of the solutions found by two algorithms were compared after running the same number of generations. The generation number varies from 100 to 1200. And all the results are the average values of 60 times running.

Figure 6 displays the fitness values of solutions for both algorithms in each test case. Before running 300 generations tradition GA has better solution. However, after that, VLCGA outperforms tradition GA continuously. This demonstrates that VLCGA can find a better solution than tradition GA after running a few generations.

Since, VLCGA involves all paths in searching, it has larger resolve space, which will lead to a higher resolve rate and a better solution. This is consistent with the experiment results.

CONCLUSION

This study proposes a novel VLCGA-based solution for multi-path QoS-aware web service composition. The variable length chromosome is used to represent the different composition plan for service composition. This approach can easily support the multi-path service composition and find the solution effectively.

Besides, VLCGA has good scalability. It support major workflow structures, new paths can be added conveniently and the re-composition is also supported by this approach.

However, current approach cannot support the gene exchange inside the And/Or workflow pattern. And the parameter matching is not adequate for universal service composition. More general structure exchanging and semantic matching should be researched for the future work. Furthermore, GA is a stochastic algorithm and the fitness of individuals in initial generation affects the performance of GA (including astringency and quality of solution). Combining this algorithm with other heuristic algorithm at initialization stage to improve the algorithm's effectiveness will also be studied in following work.

REFERENCES

Alrifai, M. and T. Risse, 2009. Combining global optimization with local selection for efficient QoS-aware service composition. Proceedings of the 18th international conference on World Wide Web, April 20-24, Madrid, Spain, pp: 881-890.

Ardagna, D. and B. Pernici, 2007. Adaptive service composition in flexible processes. IEEE Trans. Software Eng., 33: 369-384.

Beek, M., A. Bucchiarone and S. Gnesi, 2007. Web service composition approaches: From industrial standards to formal methods. Proceedings of the Internet and Web Applications and Services, May 13-19, New York, USA., pp: 15-20.

Berbner, R., M. Spahn, N. Repp, O. Heckmann and R. Steinmetz, 2006. Heuristics for QoS-aware web service composition. Proceedings of the IEEE International Conference on Web Services, Sept. 18-22, Chicago, IL, pp: 72-82.

Brie, A.H. and P. Morignot, 2005. Genetic planning using variable length chromosomes. Proceedings of the 15th Internet Conference on Automated Planning and Scheduling, (ICAPS'05), USA., pp: 320-329.

Canfora, G., M.D. Penta, R. Esposito and M.L. Villani, 2005a. An approach for QoS-aware service composition based on genetic algorithms. Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, June 25-29, ACM, New York, pp: 1069-1075.

Canfora, G., M.D. Penta, R. Esposito and M.L. Villani, 2005b. QoS-aware replanning of composite web services. Proceedings of the ICWS IEEE International Conference on Web Services, July 11-15, Washington, DC, USA., pp: 121-129.

Dong, J., 2009. A web service composition method based on multi-path. Proceedings of the 2009 IITA International Conference on Services Science, Management and Engineering, July 11-12, Washington, DC, USA., pp: 430-432.

Ericross and S. Peter, 2002. Genetic algorithm. http://en.wikipedia.org/wiki/Genetic_algorithm

Hewett, R., P. Kijsanayothin and B. Nguyen, 2009. Scalable optimized composition of web services with complexity analysis. Proceedings of the IEEE International Conference on Web Services, July 6-10, Washington, DC, USA., pp: 389-396.

Kwon, J., H. Kim, D. Lee and S. Lee, 2008. Redundant-free web service composition based on a Two-phase algorithm. Proceedings of the IEEE International Conference on Web, Sept. 23-26, Washington, DC, USA., pp: 361-368.

O'Sullivan, J., D. Edmond and A.T. Hofstede, 2004. What's in a service. Distributed Parallel Databases, 12: 117-133.

Xu, H. and T. Xu, 2008. Improved anycast routing algorithm. Comput. Eng., 34: 114-116.

Van der Aalst, W.M.P., A.H.M. ter Hofstede, B. Kiepuszewski and A.P. Barros, 2003. Workflow patterns. Distributed Parallel Databases, 14: 5-51.

Yan-Ping, C. and L. Zeng-Zhi, 2007. E-WsFrame: A framework support QoS driven web services composition. Inform. Technol. J., 6: 390-395.

Yin, K., B. Zhou, S. Zhang, H. Jiang and J. Cristoforo, 2010. Optimizing services composition in multi-network environment. Inform. Technol. J., 9: 399-411.

Zeng, L.Z., B. Boualem, A.H.H. Ngu, M. Dumas, J. Kalagnanam and H. Chang, 2004. QoS-aware middleware for web services composition. IEEE Trans. Software Eng., 30: 311-327.

Zhang, C., S. Su and J. Chen, 2006. Genetic algorithm on web service selection supporting QoS. Chinese J. Comput., 29: 1029-1037.