University of Wollongong

Research Online

Faculty of Engineering and Information
Sciences - Papers: Part A

Faculty of Engineering and Information
Sciences

1-1-2014

Impacts of pheromone modification strategies in ant colony for data-intensive service provision

Lijuan Wang

University of Wollongong, lw840@uowmail.edu.au

Jun Shen

University of Wollongong, jshen@uow.edu.au

Junzhou Luo

Southeast University

Follow this and additional works at: <https://ro.uow.edu.au/eispapers>

Part of the [Engineering Commons](#), and the [Science and Technology Studies Commons](#)

Research Online is the open access institutional repository for the University of Wollongong. For further information contact the UOW Library: research-pubs@uow.edu.au

Impacts of pheromone modification strategies in ant colony for data-intensive service provision

Abstract

In the provision of dynamic data-intensive services, the cost and response time of data sets as well as the states of services may change over time. An ant colony system for this problem is studied in this paper. Specifically, we consider changing the QoS attributes of services and replacing a certain number of services with new ones at different frequencies. In order to adapt the ant colony system to handle the dynamic scenarios, several pheromone modification strategies in reaction to changes of the optimization scenarios are investigated. The aim of the strategies is to find a balance between preserving enough old pheromone information to speed up the search process, and resetting enough new pheromone information to facilitate the ants to find a new solution for the changed scenarios. The strategies differ in their degree of reinitialized pheromone values with respect to the information that has been used to decide the amount of pheromone values. Moreover, the behaviors of different strategies for modifying pheromone information are compared.

Keywords

data, colony, ant, strategies, intensive, modification, provision, pheromone, impacts, service

Disciplines

Engineering | Science and Technology Studies

Publication Details

Wang, L., Shen, J. & Luo, J. (2014). Impacts of pheromone modification strategies in ant colony for data-intensive service provision. 21st IEEE International Conference of Web Services (ICWS 2014) (pp. 177-184). United States: Institute of Electrical and Electronics Engineers.

Impacts of Pheromone Modification Strategies in Ant Colony for Data-Intensive Service Provision

Lijuan Wang, Jun Shen

School of Information Systems and Technology
University of Wollongong, Wollongong, NSW, Australia
Email: lw840@uowmail.edu.au, jshen@uow.edu.au

Junzhou Luo

School of Computer Science and Engineering
Southeast University, Nanjing, P. R. China
Email: jluo@seu.edu.cn

Abstract—In the provision of dynamic data-intensive services, the cost and response time of data sets as well as the states of services may change over time. An ant colony system for this problem is studied in this paper. Specifically, we consider changing the QoS attributes of services and replacing a certain number of services with new ones at different frequencies. In order to adapt the ant colony system to handle the dynamic scenarios, several pheromone modification strategies in reaction to changes of the optimization scenarios are investigated. The aim of the strategies is to find a balance between preserving enough old pheromone information to speed up the search process, and resetting enough new pheromone information to facilitate the ants to find a new solution for the changed scenarios. The strategies differ in their degree of reinitialized pheromone values with respect to the information that has been used to decide the amount of pheromone values. Moreover, the behaviors of different strategies for modifying pheromone information are compared.

Index Terms—Ant colony system, data-intensive service composition, dynamic optimization.

I. INTRODUCTION

In recent years, the explosion of digital data has led to greater and greater dependence on data-intensive services. The number and complexity of data-intensive services are set to increase with an even more dynamic environment in the future. To solve a complex data handling problem, it needs to combine data from various services. In the data-intensive service composition problem, each service requests and/or creates a large amount of data sets as inputs and/or outputs, when compared with the traditional service composition problem. Each data set may be replicated at several locations, and the cost and response time of service depend on the requesters' location and the amount of transferred data. The data plays the dominant role in the data-intensive service composition problem. The service provision, and in particular the service composition, will face new challenges. First, the large number of data sets and increases of functionally equivalent services make the composition complex. Second, the size and the number of distributed data sets increase the communication and storage costs, which affect the performance of the whole application process. Third, the cost of transferring data to and from service platforms increases as the number of data sets increases. Fourth, the dynamic nature of cloud

computing and data replication needs dynamic and adaptive mechanisms to regulate the interaction between data and service users and providers.

In [1], we have presented a hierarchical taxonomy of Web service selection and composition approaches. Many methods have been applied to tackle Web service selection and composition problems, such as the local optimization methods [2], [3], the mixed integer programming (MIP) algorithms [4], [5], the heuristic-based algorithms [6], [7], and the bio-inspired algorithms [8]–[10], mainly based on genetic algorithms. By a detailed analysis of each method, we find that bio-inspired algorithms can overcome the challenging requirements of data-intensive service provision [1]. It is useful for the provision of data-intensive services to explore key features and mechanisms of biological systems and add biological mechanisms to services system. We have been applying bio-inspired algorithms to tackle the data-intensive service composition problems [11]–[17]. In the bio-inspired algorithms, the ant colony optimization algorithms can run continuously and are generally capable of adapting to changes of an optimization problem. Compared with genetic algorithms and the MIP algorithm, the ant colony optimization algorithms can find solutions more quickly in dynamic environments without restarting the optimization process, since the ants can react explicitly to the changes based on the pheromone information [1]. In this paper, we explore strategies to apply the ant colony system (ACS) to solving the dynamic data-intensive service composition problem, where services change at certain intervals.

In a dynamic environment, the service composition optimization process should be conducted repeatedly when the changes of the states of services occur, such as the changes of QoS attributes of services, the discontinuation of services, and the increase of new services. If each change in the composition is not too significant, it is likely that the solution to the changing optimization scenario will be related to the old ones to some extent. The old optimal solution can be reused to find a good solution quickly after a change occurred. A simple restart of the optimization process, which discards the former solution after a change has occurred, might not be a good strategy in most cases. On the other hand, if too much old pheromone information

is maintained, the ants will be stuck in a local optimal solution. The key is to find a trade-off between preserving old pheromone information and modifying enough new pheromone information to allow the ants to find solutions for the new search space in later iterations.

The contributions of this paper are three-fold: first, we consolidate an ant colony algorithm for the dynamic data-intensive service composition problem; second, four pheromone modification strategies in the ant colony are presented; third, we conduct experiments from two case studies with respect to the optimization behavior and the loss in the quality of the best solution, to compare different strategies. The experimental results indicate that the performance of each strategy depends not only on the strategy-specific parameter but also on the degree of changes and the frequency of occurrence of changes.

The remainder of this paper is organized as follows. Section II investigates how the ant colony algorithm is used to solve the data-intensive service composition problem, while section III describes the strategies for modifying the pheromone information to adapt the ant colony system algorithm to handle the dynamic data-intensive service composition problem. Then section IV presents experimental results and analyses. Finally, section V concludes this paper and proposes future work.

II. THE ANT COLONY SYSTEM FOR THE DATA-INTENSIVE SERVICE COMPOSITION

In this section, some basic concepts are explained. Then we only describe the general approach of ACS for our problem. The strategies added for being applied to the dynamic scenarios are presented later in section III.

In a Web service environment, abstract services are the functional descriptions of services, and concrete services represent the existing services available for potential invocation of their functionalities and capabilities. When the functions of several concrete services are consistent with the functional description of an abstract service, these concrete services are the service candidates for the abstract service and QoS attributes are used to distinguish them.

The service composition problem is modeled as a directed graph with a start vertex and an end vertex when using ACS. The data-intensive service composition problem is an extension of the traditional service composition problem, in which data sets as inputs and outputs, are incorporated. The start vertex is set as the ants' nest and the end vertex is set as the food source. The feasible solutions to the composition problem correspond to paths through the directed graph. In the directed graph, all ants are initially positioned at the start vertex and the task of each ant is to find a path from the start vertex to the end vertex. Fig. 1 shows a directed graph for the data-intensive service composition problem. In Fig. 1, $cs_{r,s}$ represents the s^{th} concrete service in the corresponding service candidate set cs_r for abstract service r .

Fig. 1: A directed graph when using ACS

ACS consists of several iterations where in the iteration each ant in the ant colony individually constructs a candidate solution for the problem. Starting at the start vertex, each ant builds up a solution iteratively by always selecting the next vertex based on pheromone trails and heuristic information. The pheromone trail, which is denoted by τ_{ij} , and the heuristic information, which is denoted by η_{ij} , are indicators of tendency to move from vertex i to vertex j . All edges of the directed graph are initialized with a certain amount of pheromone τ_0 at the start. The data-intensive service composition based on ACS is illustrated in Algorithm 1.

In Algorithm 1, q is a random number uniformly distributed in $[0, 1]$, and q_0 is a parameter ($0 \leq q_0 \leq 1$). With probability q_0 , an ant k at vertex i chooses the next vertex j from N_i^k which maximizes $[\tau_{ij}]^\alpha [\eta_{ij}]^\beta$, where parameter α is used to control the influence of τ_{ij} and parameter β is used to control the influence of η_{ij} . N_i^k is used to denote the set of vertices that have not been visited yet. When finding an executed path from the start vertex to the end vertex, the ants use a local pheromone update rule that they apply immediately after having crossed edge (i, j) . The variable ξ ($0 < \xi < 1$) is a parameter which is used to determine the local evaporation rate. During the iteration, after all ants arrive at the end vertex, a global pheromone update is performed to the best path found so far. The variable ρ ($0 < \rho < 1$) is the global pheromone evaporation rate. $Utility(S)$ is the utility of the best path S , which was described in our earlier work [13].

III. DYNAMIC ANT COLONY SYSTEM

In this section, we discuss strategies for modifying the pheromone information to adapt ACS to handle the dynamic scenarios, when new services are provided, some services are discontinued, or the QoS attributes of some services are changed. When a change of the search space of the optimization problem occurred, it is necessary to initialize the pheromone information for the new services and to modify the pheromone information for the old services that are still in provision. The solutions that were bad before a change, might be good afterwards. If ACS has converged to a path on which the amount of pheromone

Algorithm 1 The data-intensive service composition based on ACS

Initialization:

MaxIt: the maximum number of iterations;

noa: the number of artificial ants;

Input:

G: the directed graph;

Output:

S: a service execute path to create a composite service;

```

1:  $S = \emptyset$ ;  $step = 0$ ;
2: while  $step < MaxIt$  do
3:    $step = step + 1$ ;
4:   set all ants at start vertex;
5:   for each ant  $k$  in  $noa$  do
6:      $list = \emptyset$ ; //candidate list for each ant
7:     while ant  $k$  is not at the end vertex do
8:       if  $q \leq q_0$  then
9:         ant  $k$  chooses successor  $j$  which maximizes
            $[\tau_{ij}]^\alpha [\eta_{ij}]^\beta$ ;
10:      else
11:        ant  $k$  chooses successor  $j$  according to the
           probability distribution  $p_{ij}^k = \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{j \in N_i^k} [\tau_{ij}]^\alpha [\eta_{ij}]^\beta}$ ;
12:      end if
13:      update candidate list;
14:      apply the local updating rule,  $\tau_{ij} = (1 - \xi)\tau_{ij} + \xi\tau_0$ ;
15:    end while
16:  end for
17:  when all ants arrive at the end vertex, find the best
    path from all candidate lists and compare the utility
    of the best path with the utility of  $S$ . If the utility
    of the best path is larger than the utility of  $S$ , then
     $S$  is replaced by the best path.
18:  apply the global updating rule,  $\tau_{ij} = (1 - \rho)\tau_{ij} + \rho * Utility(S)$ ;
19: end while
20: return  $S$ .
```

trails becomes much higher than on all other paths, this path prevents the ants from taking another path when a change occurs on it, and the result will be a sub-optimal solution. We need to modify the pheromone information to find a balance between preserving enough old information to speed up the search process, and modifying enough new information to facilitate the ants to find a new solution for the changed scenarios.

A. Pheromone Modification Approaches

The literature presents three types of pheromone modification approaches: global pheromone modification approaches, local pheromone modification approaches, and combination of global and local pheromone modification approaches.

The global pheromone modification approaches, such as the approach to reinitializing the whole pheromone matrix, and the approach to increasing the pheromone values proportionately to their difference to the maximum pheromone value, are designed to reset all the pheromone values to a certain degree [18]–[20]. However, the global approaches do not take into account where the change of the search space actually occurred. Moreover, when we have a big search space, if a change occurs somewhere near the edge of the space, too much information might be lost by using global pheromone modification approaches.

The local pheromone modification approaches are designed to reset the pheromone values near the changes. Often, solutions to the changed scenarios will differ only locally from solutions to the old one. Therefore, the resetting of pheromone values should be performed in the close vicinity of the changes. Usually, the pheromone values are modified based on the factors contributing to an ant's local decisions. That is to say, modifying pheromone information is based on the heuristic information or the pheromone information.

The combination of the global and local pheromone modification approaches could be advantageous in a situation where local resetting is necessary to a change while global resetting is needed to facilitate the algorithm [19].

B. Pheromone Modification Strategies

The strategies we follow to modify the pheromone values are inspired by the study [19]. Suppose a change occurs at vertex $j \in cs_{cj}$, and the set of its direct successors is denoted by cs_{sj} and the set of its direct predecessors is denoted by cs_{pj} . A new parameter as the reset-value, denoted by γ_i ($\gamma_i \in [0, 1]$), is introduced to determine the amount of reinitialized pheromone values on edges adjacent to vertex i ($\forall i \in cs_{cj}$) according to (1) and (2).

$$\tau_{ik} = (1 - \gamma_i)\tau_{ik}^{old} + \gamma_i\tau_0, \quad \text{if } k \in cs_{sj} \quad (1)$$

$$\tau_{ki} = (1 - \gamma_i)\tau_{ki}^{old} + \gamma_i\tau_0, \quad \text{if } k \in cs_{pj} \quad (2)$$

The variables τ_{ik}^{old} and τ_{ki}^{old} are the pheromone values on edge (i, k) and edge (k, i) before the change occurs. In fact, pheromone values are not completely modified, but a trace of old values remains. If there is a new service, namely s , it received reset-value of $\gamma_s = 1$. We herein describe in more detail how the different strategies assign the values γ_i .

1) *R-strategy*: The first strategy is denoted as the *R*-strategy, which reinitializes all the pheromone values by the same degree. Each vertex i is assigned a strategy-specific parameter $\gamma_R \in [0, 1]$ as its reset-value, namely $\gamma_i = \gamma_R$.

2) *η -strategy*: The second strategy is denoted as the *η -strategy*, which uses the heuristic information to decide the amount of pheromone trails. Each vertex i is given a value γ_i proportionate to its utility related to vertex k . The utility U_{ik}^η and U_{ki}^η are derived from η_{ik} and η_{ki} , which are given

by (3) and (4), respectively.

$$U_{ik}^\eta = 1 - \frac{\eta_{avg}}{\gamma_E * \eta_{ik}}, \quad \eta_{avg} = \frac{\sum_{k=1}^{m_1} \eta_{ik}}{m_1}, \quad \text{if } k \in cs_{sj} \quad (3)$$

$$U_{ki}^\eta = 1 - \frac{\eta_{avg}}{\gamma_E * \eta_{ki}}, \quad \eta_{avg} = \frac{\sum_{k=1}^{m_2} \eta_{ki}}{m_2}, \quad \text{if } k \in cs_{pj} \quad (4)$$

The variables m_1 and m_2 are the number of concrete services in the sets cs_{sj} and cs_{pj} . The strategy-specific parameter $\gamma_E \in (0, \infty)$ scales the width of the utility proportion. Then γ_i is computed according to (5) and (6).

$$\gamma_i = \max\{0, U_{ik}^\eta\}, \quad \text{if } k \in cs_{sj}. \quad (5)$$

$$\gamma_i = \max\{0, U_{ki}^\eta\}, \quad \text{if } k \in cs_{pj}. \quad (6)$$

3) τ -strategy: The third strategy is denoted as the τ -strategy, which used the pheromone information to decide the reset-values. The pheromone-utility U_{ik}^τ of edge (i, k) and the pheromone-utility U_{ki}^τ of edge (k, i) are given by (7) and (8).

$$U_{ik}^\tau = \tau_{ik}, \quad \text{if } k \in cs_{sj} \quad (7)$$

$$U_{ki}^\tau = \tau_{ki}, \quad \text{if } k \in cs_{pj} \quad (8)$$

The strategy-specific parameter $\gamma_T \in (0, \infty)$ is used to limit the result to 1 for application of (1) and (2). Then γ_i is computed according to (9) and (10).

$$\gamma_i = \min\{1, \gamma_T * U_{ik}^\tau\}, \quad \text{if } k \in cs_{sj}. \quad (9)$$

$$\gamma_i = \min\{1, \gamma_T * U_{ki}^\tau\}, \quad \text{if } k \in cs_{pj}. \quad (10)$$

4) G -strategy: The fourth strategy is denoted as the G -strategy, which belongs to the global pheromone modification approach and reinitializes the whole pheromone matrix. That is to say, $\tau_{ij} = \tau_0, \forall i, j \in V$, and V is the set of vertices of the directed graph.

IV. EXPERIMENTS AND ANALYSES

A. Test Case Generation

For the purpose of our evaluation, we tested the proposed algorithm with two different case studies: one with nine abstract services as shown in Fig. 1, the other with thirty abstract services. The second case study was created by either placing a candidate set into Fig. 1 or adding another composition structure as substructure. For each directed graph, we suppose that there are twenty concrete services in each service candidate set and each concrete service requires a set of ten data sets. Then ten concrete services in each service candidate set are reserved to form a spare pool of services before the start of the algorithm, leaving each service candidate set with ten concrete services. The criteria to measure the performance of ACS are the utility of the best solution and the loss in quality of the solution compared with the MIP approach.

As mentioned in section I, the dynamical changes that can occur to the problem are the changes of QoS attributes

of services (replacement of vertices), the discontinuation of services (deletion of vertices), and the increase of new services (insertion of vertices). During the run of the algorithms, the actual scenario was changed every t iterations by replacing k services between the actual instance and the spare pool. We tested all combinations of parameter values $k \in \{1, 5, 10\}$ and $t \in \{50, 100, 200\}$. For each configuration (k, t) , 20 test runs are performed and every run was stopped after 1000 iterations. Then the average values over these 20 runs are used for comparison. When deciding which vertex to replace, the concrete service was chosen at random. This procedure continues until the number of replacing services is k .

The parameters for ACS were $\alpha = 1$, $\beta = 2$, $q_0 = 0.9$, $\rho = 0.1$, and $\xi = 0.1$. The number of ants in the colony was 10. There are two branches in the conditional structure with the probability of 0.5. The loop structure can be unfolded by cloning the vertices involved in the structure as many times as the maximal loop count [21].

For all combinations of parameters (k, t) , we tested ACS with the global pheromone modification approach and the local pheromone modification approach. For the two approaches, we compared four strategies:

- 1) R -strategy with all strategy-specific parameters $\gamma_R \in \{0.25, 0.5, 0.75, 1.0\}$.
- 2) η -strategy with all strategy-specific parameters $\gamma_E \in \{0.5, 1.0, 2.0, 5.0\}$.
- 3) τ -strategy with all strategy-specific parameters $\gamma_T \in \{0.5, 1.0, 2.0, 5.0\}$.
- 4) G -strategy with no strategy-specific parameter in it.

B. Results of Case Study One

A detailed view of the optimization behavior for each individual strategy and its strategy-specific parameter are shown in Figs. 2-5. These figures are for the case of $(k, t) = (1, 50)$, namely, small changes occur frequently. In particular, the effect on increasing the strategy-specific parameter values can be observed. After each change occurs, we see a hop of the utility of best found solution having been found in these figures. For example, after 50-th iteration a change occurs for the first time and after

Fig. 2: The optimization behavior of R -strategy with all parameters in case study one

Fig. 3: The optimization behavior of η -strategy with all parameters in case study one

Fig. 4: The optimization behavior of τ -strategy with all parameters in case study one

Fig. 5: The optimization behavior of G -strategy in case study one

100-*th* iteration a change occurs for the second time. Then the ants need to find a solution for the new scenario between 51-*th* iteration and 100-*th* iteration and let fi be the iteration where the best utility for the new scenario presents first time. Smaller value of $fi - 50$ suggests a quick recovery from the change occurred in 51-*th* iteration. From these figures, we observe that $\gamma_R = 1.0$ in R -strategy, $\gamma_E = 0.5$ in η -strategy, and $\gamma_T = 2.0$ in τ -strategy give better results than other strategy-specific parameters for the individual strategies. For other combination of k and t , the optimization behaviors for each individual strategy are similar with that in Figs. 2-5.

Besides the optimization behavior of each strategy, we

also recorded the loss in quality of the best found solution compared with the solution found by the MIP algorithm. As ACS is sub-optimal, we have evaluated the quality of the solution obtained by ACS through comparing it with the optimal solution obtained by the MIP algorithm. The global best utility of the solution obtained by the MIP algorithm is denoted by U_{global} , and the global best utility of the solution obtained by ACS is denoted by U_{acs} . Then we compute the loss in the quality, which is given by (11).

$$loss = \frac{U_{global} - U_{acs}}{U_{global}} * 100\% \quad (11)$$

For each strategy and its strategy-specific parameter, we got the average results of 20 runs of 1000 iterations. We used the open source integer programming system *lpsolve* version 5.5 [22] to implement the MIP. Fig. 6 gives a more detailed view of the performance of each strategy and its different strategy-specific parameters for the case of $(k, t) = (1, 50)$. The upper row shows the loss of the quality for R -strategy with strategy-specific parameter $\gamma_R \in \{0.25, 0.5, 0.75, 1.0\}$ from left to right, the middle two rows show the loss of the quality for η -strategy and τ -strategy with strategy-specific parameter $\gamma_E, \gamma_T \in \{0.5, 1.0, 2.0, 5.0\}$ from left to right, and the whole lower row show the loss of the quality for G -strategy since there is no strategy-specific parameter in it. The smaller value the loss is, the brighter the grid will be. Judging from the gray scale of the grid, the best utility is the R -strategy with a parameter $\gamma_R = 1.0$. The η -strategy with $\gamma_E = 0.5$ and the τ -strategy with $\gamma_T = 2.0$ are also able to achieve good solutions. The G -strategy only better than the η -strategy with $\gamma_E = 5.0$.

Figs. 7-10 show the performance of each strategy and its different strategy-specific parameters for all the cases of the combination of k and t . Judging from the “average darkness”, the best overall strategy is the R -strategy with a parameter $\gamma_R = 1.0$. The η -strategy with $\gamma_E = 2.0$ provides good solutions when $t = 100$. The τ -strategy with $\gamma_T = 5.0$ provides good solutions when $t = 200$. The G -strategy provides good solutions when there are many changes.

Fig. 6: The loss in quality of the best found solution of R -strategy, η -strategy, τ -strategy, and G -strategy for the case of $(k, t) = (1, 50)$ in case study one

Fig. 7: The loss in quality of the best found solution of R -strategy with different values of γ_R , k and t in case study one

Fig. 8: The loss in quality of the best found solution of η -strategy with different values of γ_E , k and t in case study one

Fig. 9: The loss in quality of the best found solution of τ -strategy with different values of γ_T , k and t in case study one

C. Results of Case Study Two

For case study two, the optimization behavior for each individual strategy and its strategy-specific parameter are shown in Figs. 11-14. These figures are for the case of $(k, t) = (1, 50)$. From these figures, we observe that $\gamma_R = 0.5$ in R -strategy, $\gamma_E = 0.5$ in η -strategy, and $\gamma_T = 1.0$ in τ -strategy give better results than other strategy-specific parameters for the individual strategies in

Fig. 10: The loss in quality of the best found solution of G -strategy with different values of k and t in case study one

Fig. 11: The optimization behavior of R -strategy with all parameters in case study two

Fig. 12: The optimization behavior of η -strategy with all parameters in case study two

general. Compared with the optimization behavior for each individual strategy given in Figs. 2-5, the average global utility at each interval gradually increase. That is because in case study one the ants can find the best utility in few iterations after a change occurred, but in case study two, the number of abstract services increases so the ants need more iterations to find the best solutions.

Figs. 15-18 show the performance of each strategy and its different strategy-specific parameters for all the cases of the combination of k and t . Judging from the “average darkness”, the R -strategy gives better solutions when $k =$

Fig. 13: The optimization behavior of τ -strategy with all parameters in case study two

Fig. 14: The optimization behavior of G -strategy in case study two

Fig. 15: The loss in quality of the best found solution of R -strategy with different values of γ_R , k and t in case study two

1, τ -strategy gives better solutions when $t = 200$, and G -strategy gives better solutions when $k = 5, 10$ and $t = 50$.

The results of the two case studies indicate that the performance of each strategy depends not only on the strategy-specific parameter but also on the values of k and t . In the case of a small number of changes, the R -strategy could give better solutions. In the case of many changes where the changes do not occur frequently, the τ -strategy could give better solutions. In the case of many changes where the changes occur frequently, the G -strategy could give better solutions. In our case studies, the performance of η -strategy

Fig. 16: The loss in quality of the best found solution of η -strategy with different values of γ_E , k and t in case study two

Fig. 17: The loss in quality of the best found solution of τ -strategy with different values of γ_T , k and t in case study two

Fig. 18: The loss in quality of the best found solution of G -strategy with different values of k and t in case study two

is in the middle of all strategies. A likely explanation of this fact is that the definition of η information is based on the optimization problem.

V. CONCLUSIONS

In the dynamic data-intensive service composition problem, the states of services change at certain intervals. In order to adapt the ant colony system to handle the

dynamic changes, we designed four strategies to modify the pheromone information. Two case studies were used to evaluate the proposed algorithm. For each case study we compared the performance of the four strategies with respect to the optimization behavior and the loss in quality of the best found solution. The experimental results indicate that the performance of each strategy depends not only on the strategy-specific parameter but also on the number of changes and the frequency of occurrence of changes. If there are many changes and the changes occur frequently, the global pheromone modification strategy could give better solutions. If there are many changes and the changes do not occur frequently, the local pheromone modification strategy which uses the pheromone information to decide the reset-values could give better solutions. If only a few changes occur, the local reinitialization strategy could give better solutions.

A common problem when looking at the ant colony system is the large number of parameters in the algorithm. The influence of these parameters to the strategies and performance of ant colony system is worth further investigation. Also, it needs to clarify where the boundary exactly lies between the local modification approach and the global modification approach.

ACKNOWLEDGMENT

This work is supported by National Key Basic Research program of China under Grants No. 2010CB328104, National Natural Science Foundation of China under Grants No. 61320106007 and No. 61202449, China National High Technology Research and Development Program under Grants No. 2013AA013503, China Specialized Research Fund for the Doctoral Program of Higher Education under Grants No. 20110092130002, Jiangsu Provincial Key Laboratory of Network and Information Security under Grants No. BM2003201, and Key Laboratory of Computer Network and Information Integration of Ministry of Education of China under Grants No. 93K-9.

REFERENCES

- [1] L. Wang and J. Shen, "A Critical Systematic Review of Service Concretization Based on Bio-Inspired Approaches," 2014, <http://ro.uow.edu.au/eisapers/1903>.
- [2] B. Benatallah, M. Dumas, Q. Z. Sheng, and A. H. H. Ngu, "Declarative Composition and Peer-to-Peer Provisioning of Dynamic Web Services," in *Proceedings of 18th International Conference on Data Engineering (ICDE '02)*. Washington, DC, USA: IEEE Computer Society, 2002, pp. 297–308.
- [3] F. Li, F. Yang, K. Shuang, and S. Su, "Q-Peer: A Decentralized QoS Registry Architecture for Web Services," in *Service-Oriented Computing - ICSOC 2007*, ser. Lecture Notes in Computer Science, B. Krämer, K. J. Lin, and P. Narasimhan, Eds. Berlin, Heidelberg: Springer-Verlag, 2007, vol. 4749, pp. 145–156.
- [4] D. Ardagna and B. Pernici, "Adaptive Service Composition in Flexible Processes," *IEEE Transactions on Software Engineering*, vol. 33, no. 6, pp. 369–384, 2007.
- [5] Y. Qu, C. Lin, Y. Wang, and Z. Shan, "QoS-aware Composite Service Selection in Grids," in *Proceedings of 5th International Conference on Grid and Cooperative Computing (GCC '06)*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 458–465.
- [6] P. N. Bless, D. Klabjan, and S. Y. Chang, "Heuristics for Automated Knowledge Source Integration and Service Composition," *Computers & Operations Research*, vol. 35, no. 4, 2008.
- [7] R. Berbner, M. Spahn, N. Repp, O. Heckmann, and R. Steinmetz, "Heuristics for QoS-Aware Web Service Composition," in *International Conference on Web Services (ICWS '06)*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 72–82.
- [8] G. Canfora, M. Penta, R. Esposito, and M. L. Villani, "An Approach for QoS-aware Service Composition based on Genetic Algorithms," in *Proceedings of the 2005 conference on Genetic and evolutionary computation (GECCO '05)*. New York, NY, USA: ACM, 2005, pp. 1069–1075.
- [9] Y. Ma and C. Zhang, "Quick Convergence of Genetic Algorithm for QoS-driven Web Service Selection," *Computer Networks*, vol. 52, no. 5, pp. 1093–1104, 2008.
- [10] Z. J. Wang, Z. Z. Liu, X. F. Zhou, and Y. S. Lou, "An Approach for Composite Web Service Selection Based on DGQoS," *The International Journal of Advanced Manufacturing Technology*, vol. 56, no. 9-12, pp. 1167–1179, 2011.
- [11] J. Shen, G. Beydoun, S. Yuan, and G. Low, "Comparison of Bio-Inspired Algorithms for Peer Selection in Services Composition," in *Proceedings of the 2011 IEEE International Conference on Services Computing*. Washington, DC, USA: IEEE Computer Society, 2011, pp. 250–257.
- [12] L. Wang and J. Shen, "Towards Bio-inspired Cost Minimisation for Data-intensive Service Provision," in *IEEE First International Conference on Services Economics*. Washington, DC, USA: IEEE Computer Society, 2012, pp. 16–23.
- [13] L. Wang, J. Shen, and G. Beydoun, "Enhanced Ant Colony Algorithm for Cost-Aware Data-Intensive Service Provision," in *IEEE Ninth World Congress on Services*. Washington, DC, USA: IEEE Computer Society, 2013, pp. 227–234.
- [14] L. Wang and J. Shen, "Economical Data-Intensive Service Provision Supported with a Modified Genetic Algorithm," in *IEEE 2nd International Congress on Big Data*. Washington, DC, USA: IEEE Computer Society, 2013, pp. 358–365.
- [15] L. Wang, J. Luo, J. Shen, and F. Dong, "Cost and Time Aware Ant Colony Algorithm for Data Replica in Alpha Magnetic Spectrometer Experiment," in *IEEE 2nd International Congress on Big Data*. Washington, DC, USA: IEEE Computer Society, 2013, pp. 254–261.
- [16] L. Wang, J. Shen, C. Di, Y. Li, and Q. Zhou, "Towards Minimizing Cost for Composite Data Intensive Services," in *Proceeding of the IEEE 17th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*. Washington, DC, USA: IEEE Computer Society, 2013, pp. 293–298.
- [17] L. Wang, J. Shen, and J. Yong, "A survey on bio-inspired algorithms for web service composition," in *IEEE 16th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*. Washington, DC, USA: IEEE Computer Society, 2012, pp. 569–574.
- [18] C. J. Eyckelhof and M. Snoek, "Ant Systems for a Dynamic TSP - Ants caught in a traffic jam," in *Proceedings of ANTS 2002 - From Ant Colonies to Artificial Ants: Third International Workshop on Ant Algorithms*, ser. Lecture Notes in Computer Science, M. Dorigo, G. Caro, and M. Sampels, Eds., vol. 2463. Berlin, Heidelberg: Springer-Verlag, 2002, pp. 88–99.
- [19] M. Guntsch and M. Middendorf, "Pheromone Modification Strategies for Ant Algorithms Applied to Dynamic TSP," in *Applications of Evolutionary Computing*, ser. Lecture Notes in Computer Science, E. J. W. Boers, Ed. Berlin, Heidelberg: Springer-Verlag, 2001, vol. 2037, pp. 213–222.
- [20] —, "Applying Population Based ACO to Dynamic Optimization Problems," in *Ant Algorithms*, ser. Lecture Notes in Computer Science, M. Dorigo, G. Caro, and M. Sampels, Eds. Berlin, Heidelberg: Springer-Verlag, 2002, vol. 2463, pp. 111–122.
- [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*, vol. 30, no. 5, pp. 311–327, 2004.
- [22] M. Berkelaar, K. Eikland, and P. Notebaert, "Ipsolve: Open Source (mixed-integer) Linear Programming System," 2004, <http://lpsolve.sourceforge.net/>.