University of Wollongong

Research Online

Faculty of Engineering and Information Sciences - Papers: Part A

Faculty of Engineering and Information Sciences

1-1-2014

Multi-phase ant colony system for multi-party data-intensive service provision

Lijuan Wang
University of Wollongong, lw840@uowmail.edu.au

Jun Shen
University of Wollongong, jshen@uow.edu.au

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

 Part of the Engineering Commons, and the Science and Technology Studies Commons

Recommended Citation

Wang, Lijuan and Shen, Jun, "Multi-phase ant colony system for multi-party data-intensive service provision" (2014). *Faculty of Engineering and Information Sciences - Papers: Part A*. 4090.
https://ro.uow.edu.au/eispapers/4090

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

Multi-phase ant colony system for multi-party data-intensive service provision

Abstract

The rapid proliferation of enormous sources of digital data has led to greater dependence on data-intensive services. Each service may actually request or create a large amount of data sets. To compose these services will be more challenging. Issues such as autonomy, scalability, adaptability, and robustness, become difficult to resolve. In order to automate the process of reaching an agreement among service composers, service providers, and data providers, an ant-inspired negotiation mechanism is considered in this paper. We exploit a group of agents automatically negotiating to establish agreeable service contracts. Twostage negotiation procedures are used in our data-intensive service provision model, which will provide effective and efficient service selection for service composers. We also present a multi-phase, multi-party negotiation protocol, where the ant colony system is applied to select services with the best or near-optimal utility outputs. In order to adapt the ant colony system to handle the dynamic scenarios during negotiations, we also discuss several strategies for modifying the pheromone information in the first place. The experimental results show that our negotiation-based approach can facilitate the data-intensive service provision with better outcome.

Keywords

service, provision, data, party, system, colony, intensive, ant, multi, phase

Disciplines

Engineering | Science and Technology Studies

Publication Details

Wang, L. & Shen, J. (2014). Multi-phase ant colony system for multi-party data-intensive service provision. IEEE Transactions on Services Computing, 9 (2), 264-276.

This journal article is available at Research Online: https://ro.uow.edu.au/eispapers/4090

Multi-Phase Ant Colony System for Multi-Party Data-Intensive Service Provision

Lijuan Wang, Jun Shen
School of Information Systems and Technology
University of Wollongong, NSW, Australia
E-mail: lw840@uowmail.edu.au, jshen@uow.edu.au

Abstract—The rapid proliferation of enormous sources of digital data has led to greater dependence on data-intensive services. Each service may actually request or create a large amount of data sets. To compose these services will be more challenging. Issues such as autonomy, scalability, adaptability, and robustness, become difficult to resolve. In order to automate the process of reaching an agreement among service composers, service providers, and data providers, an ant-inspired negotiation mechanism is considered in this paper. We exploit a group of agents automatically negotiating to establish agreeable service contracts. Two-stage negotiation procedures are used in our data-intensive service provision model, which will provide effective and efficient service selection for service composers. We also present a multi-phase, multi-party negotiation protocol, where the ant colony system is applied to select services with the best or near-optimal utility outputs. In order to adapt the ant colony system to handle the dynamic scenarios during negotiations, we also discuss several strategies for modifying the pheromone information in the first place. The experimental results show that our negotiation-based approach can facilitate the data-intensive service provision with better outcome.

Index Terms—Ant colony system, data-intensive service composition, automated negotiation, quality of service

1 Introduction

In the past decade, Web service selection and composition have been attracting much research attention. To obtain a viable business model for composite services, the dynamic service price-setting models are increasingly negotiation-based [1]–[5]. Meanwhile, the rapid proliferation of enormous sources of digital data has led to greater dependence on data-intensive services. Each service may request or create a large amount of data sets. Applications based on data-intensive services have become one of the most challenging applications in service oriented computing and cloud computing. A survey of the challenges, techniques, and technologies of data-intensive applications was presented in [6]. The service provision, and in particular, service composition, will face new challenges as the services and data grow drastically. Firstly, the large number of data sets and increases of functionally equivalent services make the composition complex. Secondly, the size and the number of distributed data sets increase the communication cost and the storage cost, which affect the performance of the whole application process. Thirdly, the dynamic nature of cloud computing and data replication needs dynamic and adaptive mechanisms to regulate the interactions between data and service users and providers. Indeed, new mechanisms are needed to overcome the challenges and to keep the quality attributes of composite services acceptable.

One of the motivations of our work is the Alpha Magnetic Spectrometer (AMS) experiment, which uses cloud computing to process a huge amount of data. The AMS, also designated AMS-02, is a particle physics experiment module that is mounted on the International Space Station. The purpose of the AMS experiment is to advance knowledge of the universe and lead to the understanding of its origin by searching for antimatter and dark matter while performing precision measurements of cosmic rays composition and flux. The key technology for accessing the data collected from AMS relies on data services. The AMS-02 SOC (Science Operation Center) at Southeast University in China (labeled as AMS-02 SOC@SEU) is supported by the IBM-sponsored Cloud Computing Center with 3500 CPU core and 500TB storage. The AMS-02 SOC@SEU typically receives 200GB of data from AMS and generates 700GB of data after processing them, on each day. Scientists and remote users deploy different processes, such as data mining, image processing, or data query on a large amount of data at AMS-02 SOC. The use of Web services technologies provides valuable solutions to speed up the scientific data analysis. A composition of a set of services as a composite service can be reused by other researchers.

In general, the data-intensive service composition will be cooperatively supported by three stakeholders: the service composers, the service providers, and the data providers. Providers need an approach to regulate and price their resources, either the services or data or their combination. They all want to have

a good market position maximizing their profits or utility. The decisions of all three stakeholders depend on each other. The data provider sells data sets to multiple service providers in order to maximize the data usage and the profit. The cost and response time of data sets for one service provider are affected by the demand of the others. The service providers also play the requester role with respect to the data sets. Thus, the service providers will have two aims, one is to lower the access cost and response time of data sets and the other is to maximize profit and their service usage. Also, the service providers compete with other service providers to initiate or maintain a contract with the service composers and are invariably interested in cost saving. The actual usage of services typically encourages the composers to have a long term contract with the service providers. They also select services that best match the quality of service (QoS) requirements. Meanwhile, data-intensive services are typically used in a dynamic and changing environment, and different providers typically have conflicting objectives. This paper is trying to automate the process of reaching an agreement in data-intensive service composition, which uses a group of agents automatically negotiating to establish agreeable service contracts.

The lifetime of our problem framework is described in Fig. 1. The first step is that the service composer tries to select a set of service candidates while the data provider provides data sets, and the second step is that if a feasible solution which satisfies the service composer's local and global QoS constraints does not exist, negotiations are performed in order to determine new quality values for each involved service. In the lifetime, two-stage negotiation processes are

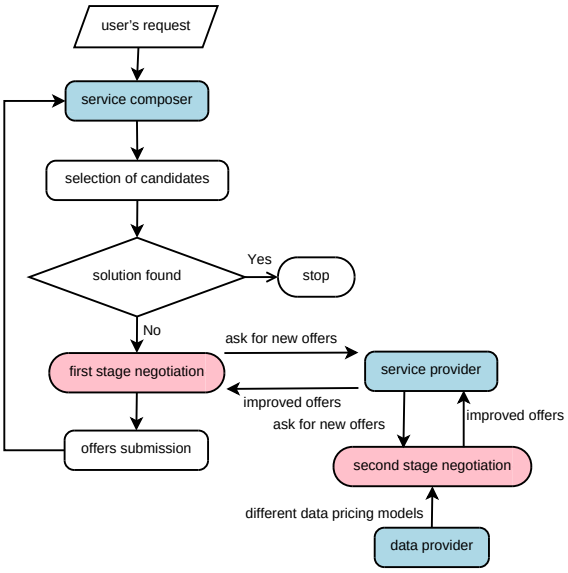


Fig. 1: The lifetime of the data-intensive service provision

used. In the first stage, a service composer negotiates with multiple service providers over each service in a structured one-to-many negotiation process. In the second stage, each service provider negotiates with a data provider over a set of data sets in a structured one-to-one negotiation process. This paper will focus on the two-stage negotiation procedures.

The ant colony optimization algorithms are inspired by the foraging behavior of ant colonies, in which a set of artificial ants cooperate to find a solution of a problem by exchanging information via pheromone deposited on a graph edges. The ant colony optimization algorithms iteratively performs a loop constitutes the ants' solution construction and the pheromone update. We apply an ant colony system (ACS) to select services in the negotiation mechanism.

During negotiations 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. Hence, we need to investigate how the ant colony system is applied to better facilitate the dynamic data-intensive service provision.

The main contributions of this paper are fivefold: first, the lifetime of the data-intensive service provision and the two-stage negotiation procedures are described; second, an ant colony system for the dynamic service composition problem is consolidated; third, four pheromone modification strategies in the ant colony system are presented; fourth, a multi-phase, multi-party negotiation protocol is designed; fifth, we conduct experiments from two case studies to compare different pheromone modification strategies and we also evaluate the negotiation-based approach in different scenarios.

The remainder of this paper is organized as follows. Section 2 introduces the related work. Section 3 investigates how an ant colony system is used to solve the data-intensive service composition problem. Section 4 describes the strategies for modifying the pheromone information to adapt the ant colony system to handle the dynamic scenarios. Section 5 proposes a multi-phase, multi-party negotiation protocol. Section 6 dis-

cusses the decision making model. Section 7 presents the experimental results and analyses. Finally, section 8 concludes this paper and proposes future work.

2 Related Work

The literature presents many methods for Web service selection and composition problems. The commonly used methods are the mixed integer programming (MIP) algorithms [7], [8], the heuristic-based algorithms [9], [10], and the bio-inspired algorithms such as the ant colony optimization algorithms [11], [12] and the genetic algorithms [13], [14]. The MIP algorithms can provide optimal solutions, but the scalability and computation time would increase when the QoS requirements become excessive or the size of the problem is very large. The heuristic-based algorithms and the bio-inspired algorithms are sub-optimal. There is a trade-off between computation cost and quality of the solution in the case of the sub-optimal methods. In [15], we presented a hierarchical taxonomy of service selection and composition methods. By a detailed analysis of each method with respect to their optimality, their computational efficiency, and their dynamic complexity, we found that bio-inspired algorithms could overcome the new challenging requirements of the typical data-intensive service provision. It is useful for the provision of data-intensive services to explore key features and mechanisms of biological systems and to add biological mechanisms to services. For example, the authors of [16] already proved that it was useful for service management and discovery to add biological mechanisms to services. Another study explained that bio-inspired computing was one of the underlying techniques to solve data-intensive problems [6]. In addition, we have conducted a comprehensive review of applications of ant colony optimization algorithms, genetic algorithms, and particle swarm optimization algorithms to service selection and composition problems [15].

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 MIP algorithms, 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. We have been applying bio-inspired algorithms to tackle the data-intensive service composition problems [17]–[22]. In [22], an economical model and an extensible QoS model were presented. In the extensible QoS model, we analyzed the effect of the data size, the data transfer time, and the data accessing time, since the communication cost of mass data transfer affects the performance of the data-intensive service application [23]. We have applied ACS to solve the data-intensive service composition problem [18]. However, in our earlier study, we did not consider scenarios where the global QoS constraints are significantly severe. On the other hand, we need to explore strategies to apply ACS to solving the dynamic scenarios where services change at certain intervals.

Negotiation has been adopted in service provision in order to get better QoS attributes. A negotiation process is the interplay of offers and counter-offers between a buyer and a seller, with different criteria and goals, working to identify a mutually acceptable solution. Automated negotiations normally follow negotiation protocols, exchange negotiation objects, and are driven by decision making models [24]. Negotiation protocols govern the negotiation by defining rules such as when the negotiation process ends, what deals can be made, and what sequences of offers are allowed. Negotiation objects are the issues such as price and time over which the negotiation takes place. Decision making models are used for evaluating and generating offers and counter-offers.

An iterative negotiation approach for a service composition was presented in [5]. The aim of the approach was to select services for the service-based systems in the scenarios where the QoS constraints were severe. In [25], the service level agreements for a service composition were established through autonomous agent negotiation. A new negotiation protocol was also presented to support coordinated negotiation. A framework was presented to select composite services by a multi-agent system using a case-based reasoning method [26], [27]. The authors of [8] introduced an approach for the Web service selection problem with large scale processes and severe QoS constraints. The Web service selection problem was formalized as a MIP problem and the loop peeling was adopted for optimization in that paper.

However, the negotiation approaches in the above studies are not able to effectively solve the data-intensive service provision problems, in which the data plays the dominant role. For example, the cost and response time of services largely depend on the accessing cost and response time of data sets. The negotiation for the data-intensive service provision is a multi-phase, multi-party process. The service composer should be able to negotiate with multiple service providers over each abstract service, while a service provider should be able to negotiate with the data providers over the data sets required by the services. To address the above issues, this paper presents an ant-inspired negotiation approach.

3 An Ant Colony System for Data-Intensive Service Composition

The data-intensive service composition problem is an extension of the traditional service composition

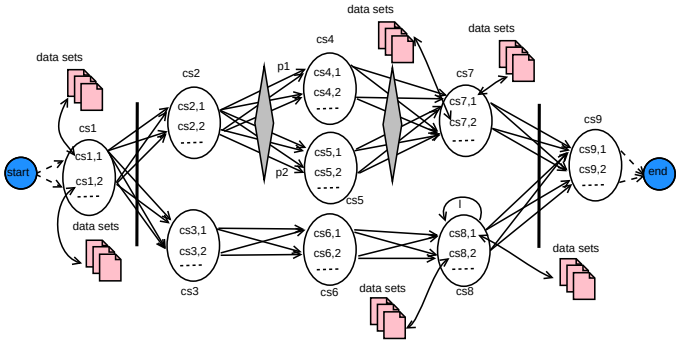


Fig. 2: A graph for the data-intensive service composition problem

problem, in which data sets as inputs and outputs of services, are incorporated. The problem is modeled as a graph with a $start$ vertex and an end vertex. 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 graph. In the 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. 2 shows a graph for our problem, in which $cs_{r,s}$ represents the s-th concrete service in the corresponding service candidate set cs_r for abstract service r.

In this section, we describe the general approach of ACS. ACS is an algorithm inspired by the ant system but differs from it in three main aspects [28]. First, the state transition rule provides a way to balance between the exploration of new edges and the exploitation of accumulated knowledge about the problem. Second, a local updating rule is applied while ants construct a path. Third, the global updating rule is applied only to edges which belong to the best ant path. The new strategies applied to the dynamic scenarios will be presented later in section 4.

3.1 State Transition Rule

In ACS, when ant k arrived at vertex i, it chooses a vertex j to move to by applying the rule given by (1).

$$j = \begin{cases} \arg\max_{j \in N_i^k}\{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta\}, & \text{if } q \leq q_0; \\ \text{selected according to } p_{ij}^k, & \text{otherwise.} \end{cases} \quad (1)$$

where

$$p_{ij}^k = \begin{cases} \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{j \in N_i^k} [\tau_{ij}]^\alpha [\eta_{ij}]^\beta}, & \text{if } j \in N_i^k \ ; \\ 0, & \text{otherwise.} \end{cases}$$

The variable τ_{ij} is the pheromone trail of edge (i, j), η_{ij} is the heuristic information and it is set as the quality utility of the service at vertex j. α and β are parameters to control the influence of τ_{ij} and η_{ij} respectively. The pheromone trail and the heuristic information are indicators of tendency to move from vertex i to vertex j. q is a random number uniformly distributed in $[0, 1]$, q_0 is a parameter ($0 \leq q_0 \leq 1$) which determines the relative importance of exploration versus exploitation. In other words, with probability q_0 the ant exploits the learned pheromone trails, while with probability $(1 - q_0)$ it performs a biased exploration of the arcs [28]. N_i^k is used to denote the set of vertices that have not been visited yet. If $q > q_0$, j is randomly selected according to the probability distribution p_{ij}^k.

3.2 Local Updating Rule

While building a solution to the problem, i.e., a path though the graph, the ants change the pheromone level on visited edges by applying a local updating rule of (2).

$$\tau_{ij} = (1 - \xi)\tau_{ij} + \xi\tau_0 \quad (2)$$

The variable ξ $(0 < \xi < 1)$ is used to determine the local evaporation rate, and τ_0 is the initial pheromone level.

3.3 Global Updating Rule

In each iteration, after all ants arrived at the end vertex, a global pheromone updating rule is performed to the best path found so far, which is given by (3).

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \rho\Delta\tau_{ij}, \forall(i, j) \in S \quad (3)$$

The variable ρ $(0 < \rho < 1)$ denotes the global pheromone evaporation rate. The variable $\Delta\tau_{ij} = U(S)$ is the overall quality utility of the best path S, which is computed according to the QoS aggregation function based on the quality utility of the selected services. The simple additive weighting approach was adopted for the utility function. We have proposed an extensible QoS model for the data-intensive services, which included the access cost and response time of data sets [22]. ACS stops when a termination condition is true, for example, a maximum number of iterations has been done. The data-intensive service composition based on ACS is illustrated in Algorithm 1. Our algorithm is an empirical refinement of ACS which can be applied recursively. We implemented it as a function which can be invoked by other procedures.

4 Pheromone Modification Strategies for Dynamic Ant Colony System

In this section, we discuss strategies for modifying the pheromone information to adapt ACS to handle the dynamic scenarios during negotiations. When a change of the search space 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. In

Algorithm 1 ACS(G); a function of ACS

Initialization:
$MaxIt$: the maximum number of iterations;
noa: the number of artificial ants;
Input:
G: the graph;
Output:
S: the best path;
$optimal$: a variable used to indicate whether S is a feasible solution;

```
S = ∅; optimal = 0; step = 0;
while step < MaxIt do
    step = step + 1;
    set all ants at the start vertex;
    for each ant k in noa do
        list_k = ∅; //candidate list for each ant
        while ant k is not at the end vertex do
            ant k chooses a successor according to (1);
            ant k updates its candidate list list_k;
            ant k applies the local updating rule (2);
        end while
    end for
    when all ants arrive at the end vertex, find the
    best path L from all candidate lists;
    if U(L) > U(S) then
        set L to S;
    end if
    apply the global updating rule (3) to S;
end while
if S is a feasible solution then
    optimal=1;
end if
return S and optimal.
```

the dynamic scenarios, solutions that were bad before a change, might be good afterwards. If ACS has converged to a path on which the amount of pheromone 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 scenario.

4.1 Pheromone Modification Approaches

The literature presents three types of pheromone modification approaches: global approaches, local approaches, and combined 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 [29], [30]. 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, the pheromone information is modified based on the heuristic information or the pheromone information.

The combined 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.

4.2 Pheromone Modification Strategies

The strategies we follow to modify the pheromone values are inspired by the study [30]. Suppose a change occurs at vertex $j \in cs_{cj}$, 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 (4) and (5).

$$\tau_{ik} = (1-\gamma_i)\tau_{ik}^{old} + \gamma_i\tau_0, \quad \text{if } k \in cs_{sj} \tag{4}$$

$$\tau_{ki} = (1-\gamma_i)\tau_{ki}^{old} + \gamma_i\tau_0, \quad \text{if } k \in cs_{pj} \tag{5}$$

We describe in more detail how the different strategies assign γ_i in the following subsections.

4.2.1 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 $\gamma_i = \gamma_R$.

4.2.2 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, according to (6) and (7).

$$\gamma_i = \max\{0, U_{ik}^{\eta}\}, \quad \text{if } k \in cs_{sj} \tag{6}$$

$$\gamma_i = \max\{0, U_{ki}^{\eta}\}, \quad \text{if } k \in cs_{pj} \tag{7}$$

where

$$U_{ik}^{\eta} = 1 - \frac{\eta_{avg}}{\gamma_E * \eta_{ik}}, \quad \eta_{avg} = \frac{\sum_{k=1}^{m_1} \eta_{ik}}{m_1}$$
$$U_{ki}^{\eta} = 1 - \frac{\eta_{avg}}{\gamma_E * \eta_{ki}}, \quad \eta_{avg} = \frac{\sum_{k=1}^{m_2} \eta_{ki}}{m_2}$$

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.

4.2.3 τ-strategy

The third strategy is denoted as the τ-strategy, which uses the pheromone information to decide the reset-values, according to (8) and (9).

$$\gamma_i = \min\{1, \gamma_T * \tau_{ik}\}, \quad \text{if } k \in cs_{sj} \tag{8}$$
$$\gamma_i = \min\{1, \gamma_T * \tau_{ki}\}, \quad \text{if } k \in cs_{pj} \tag{9}$$

The strategy-specific parameter $\gamma_T \in (0, \infty)$ is used to limit the result to 1.

4.2.4 G-strategy

The fourth strategy is denoted as the G-strategy, which belongs to the global approaches and reinitializes the whole pheromone matrix. That is to say, $\gamma_i = 1$, $\forall i, k \in V$, and V is the set of vertices of the graph.

5 The Extended Negotiation Protocol

The negotiators, namely the service composer, the service provider, and the data provider will be represented by agents. We refer to agents acting on behalf of service composers as composer agents (CAs), agents representing service providers as service agents (SAs), and agents representing data providers as data agents (DAs). Basically, an agent starts by offering a value for the negotiation object (say, a price) to its opponent. The opponent can accept the offer, or exit the negotiation, or make a counter-offer. The negotiation may be iterative as several rounds of offers and counter-offers will occur until one of the agents accepts, or exits the negotiation. The agent might exit the negotiation because the time deadline is reached without an agreement being in place. The iteration is referred to as a negotiation round, and the time deadline refers to the number of allowed rounds.

We have designed a multi-phase, multi-party negotiation protocol for our problem. The proposal is based on the iterated contract net interaction protocol, which is specified by the Foundation of Intelligent Physical Agents (FIPA) [31]. The FIPA protocol is the most widely used negotiation protocol for one-to-many negotiation in the agent community [25], [32]. It is also applicable in one-to-one negotiation, as it is considered to be a particular case of one-to-many negotiation. The protocol supports recursive negotiation and allows multi-round iterative negotiation to reach an agreement. However, the FIPA standard protocol is insufficient in supporting QoS negotiation for our problem, as it is unable to allow multi-party negotiation or combine multiple negotiation processes. It is hence extended in this paper to allow a CA to aggregate results of the individual services and to perform the confirmation to instruct SAs in the negotiations. Also, the extended protocol should allow the negotiations between SAs and DAs.

There are three phases in the extended protocol:

1) Phase 1, which allows us to find out how many SAs are available to enter the negotiation, and their offers over the objects to be negotiated upon. This phase includes one-shot interaction between the CA and all SAs, and it also includes one-shot interaction between all SAs and the DA. The CA is involved in simultaneous negotiations with multiple SAs. Each negotiation between the CA and a SA is private and holds a lot of information. Each SA is unaware of its competitor's status in the current composition. If the CA sends the current winning offers to potential SAs, SAs can analyze their positions and adapt quickly [5], [33]. This in turn significantly reduces the search space by guiding the negotiation process to proceed in the right direction towards optimal solutions. The SAs process the request from the CA and may decide to negotiate with the DA. Before a SA answers the CA, it evaluates the offers received from the DA. The DA may accept the counter-offer from a SA, or may refuse to participate in the negotiation, or may provide a new offer to the SA. Thus, the SA can answer the CA based on the results of the nested negotiation offers. The negotiation protocol is shown in Fig. 3.
2) Phase 2, which allows us to iterate the negotiation. In case the first negotiation phase ends up with counter-proposals from SAs, the CA

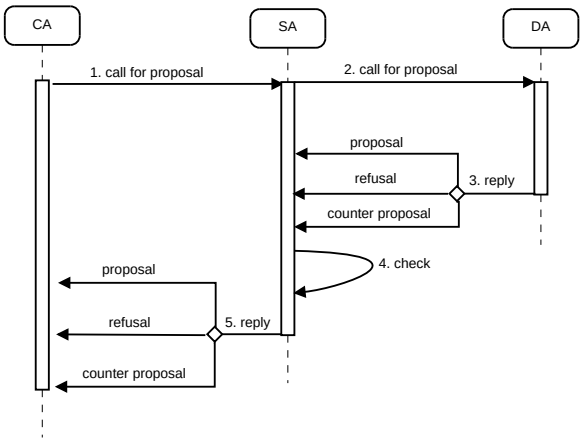


Fig. 3: Protocol to support first phase of negotiation

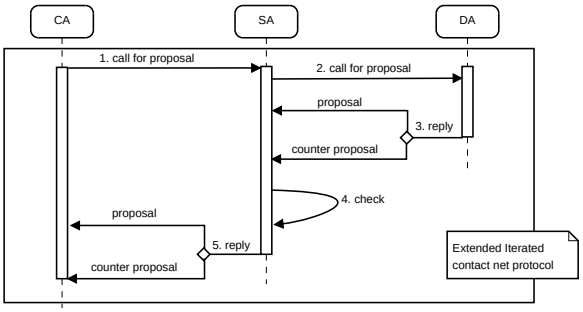


Fig. 4: Protocol to support second phase of negotiation

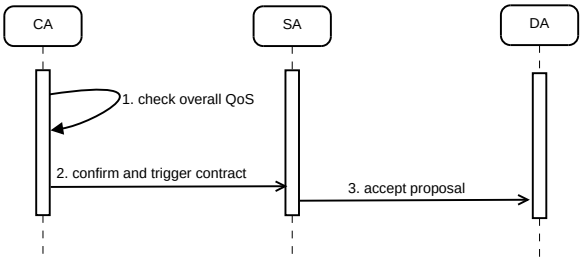


Fig. 5: Protocol to support third phase of negotiation in case of successful negotiation

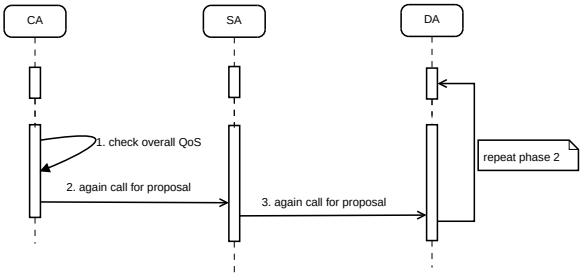


Fig. 6: Protocol to support third phase of negotiation in case of failed negotiation

will initiate negotiation with the SAs and send them new proposals with the set of current winning offer and counter-offer as its content. This negotiation phase is repeated until there is no counter-proposal from SAs or the time deadline is reached. The negotiation protocol is shown in Fig. 4.

3) Phase 3, which allows us to end the negotiation. If the second negotiation phase ends up with proposals, the CA will evaluate the overall QoS attributes. If the overall QoS requirements are satisfied, the CA informs all SAs about the acceptance of the offers. Then SAs inform the DA about the acceptance of the dependent offers. The negotiation protocol is shown in Fig. 5. In the case that the overall QoS requirements are not satisfied based on the current deals, the CA will restart the second negotiation phase with respect to one or many abstract services. Then each corresponding SA sends proposals to the DA and commences a new negotiation process. The negotiation protocol is shown in Fig. 6.

6 Decision Making Model

The negotiation process between two agents is a bilateral interaction that consists of a succession of offers and counter-offers. Let a ($a \in \{CA, SA, DA\}$) represent the negotiation agent and o ($o \in \{1, 2, \ldots, r\}$) the negotiation object. Each agent has a defined delimited range to consider the value of an object. For example, a CA has the maximum price at which it would buy the service and a SA has the minimum price at which it would sell the service. Let $x_o \in [min_o^a, max_o^a]$ be a value for object o acceptable to agent a. The agent acts alternately, making or accepting offers and counter-offers on the value x_o, or abandoning the negotiation when the time deadline is reached without an agreement being in place. Each agent a has a utility function $U_a(x_o) : [min_o^a, max_o^a] \rightarrow [0, 1]$, representing its satisfaction of the value x_o. For convenience, the utility values are kept in the interval $[0, 1]$. In this paper, we adopted linear utility function as used in many cost-benefit models in the literature [4], [34].

To negotiate on multiple object (say r), a normalized weight w_o^a ($\sum_{o=1}^{r} w_o^a = 1$) representing the relative importance of each object o for agent a is assigned under negotiation. Then an agent's utility function for an offer $x = (x_1, x_2, \ldots, x_r)$ in the multi-dimensional space is given by (10).

$$U_a(x) = \sum_{o=1}^{r} w_o^a * U_a(x_o) \tag{10}$$

We use the variable $x_{b\,!\,a}^t$ to denote the offer x that agent a received from agent b at time t, and a's interpretation of $x_{b\,!\,a}^t$ at time t^0 ($t < t^0$) is given by (11).

$$I^a(t^0, x_{b\,!\,a}^t) = \begin{cases} reject & \text{if } t^0 > t_{max}^a, \\ accept & \text{if } U_a(x_{a\,!\,b}^{t'}) \leq U_a(x_{b\,!\,a}^t), \\ x_{a\,!\,b}^{t'} & \text{otherwise.} \end{cases} \tag{11}$$

The variable $x_{a\,!\,b}^{t'}$ is used to denote the offer that agent a is ready to send to agent b at time t^0, and t_{max}^a is used to represent the time by which agent a must have completed the negotiation.

In order to prepare a counter-offer $x_{a\,!\,b}^{t'}$, agent a uses a set of tactics that generate new values for each object in the negotiation set. The time dependent tactics, which use time-based decision functions $\alpha_o^a(t)$ ($0 \leq \alpha_o^a(t) \leq 1$), are often used in negotiation systems [4], [34]–[36]. In this case, the counter-offer with respect to each object o, is generated according

to (12).

$$x_{a\ !\ b}^{t'}[o] = \begin{cases} min_o^a + \alpha_o^a(t)(max_o^a - min_o^a) \\ \qquad \text{if } U_a(x(o)) \text{ is decreasing,} \\ max_o^a - \alpha_o^a(t)(max_o^a - min_o^a) \\ \qquad \text{if } U_a(x(o)) \text{ is increasing.} \end{cases} \quad (12)$$

The time-based decision function $\alpha_o^a(t)$ can be defined as the exponential function, polynomial function, and sigmoid function [4], [34], [36]. For example, the polynomial function can be given by (13).

$$\alpha_o^a(t) = \kappa_o^a + (1 - \kappa_o^a) * \left(\frac{\min(t, t_{max}^a)}{t_{max}^a}\right)^{\frac{1}{\delta}} \quad (13)$$

The variable δ is a parameter used to control the concession rate, and κ_o^a is the initial concession at $t = 0$, where $\alpha_o^a(0) = \kappa_o^a$ $(0 \leq \kappa_o^a \leq 1)$ and $\alpha_o^a(t_{max}^a) = 1$.

For each bilateral negotiation between the CA and a SA or between a SA and the DA, an agreement is possible if there is some degree of intersection between the reservation intervals of the two agents. After CA and SA finish their negotiation process, we need to modify the pheromone information of ACS. The selection of the pheromone modification strategy depends on the negotiation results. The ant-inspired negotiation approach is illustrated in Algorithm 2.

Algorithm 2 Ant-inspired negotiation approach

Initialization:
$Tmax$: the maximum number of negotiation rounds;
Input:
G: a graph;
Output:
S and $optimal$;

1: $S = \emptyset$;
2: $iteration = 0$;
3: $[optimal, S] = ACS(G)$;
4: **while** $optimal == 0$ **do**
5: $iteration = iteration + 1$;
6: **if** $iteration > Tmax$ **then**
7: break;
8: **end if**
9: **for** each abstract service **do**
10: find out SAs that are available to enter the negotiation;
11: **for** each concrete service **do**
12: CA sends a counter-offer to SA;
13: SA sends a counter-offer to DA;
14: DA interprets the counter-offer from SA;
15: SA interprets the counter-offer from CA based on the nested offers from DA;
16: steps 12-15 are repeated until no counter-offer is proposed within the time deadline;
17: **if** a change occurs **then**
18: modify the related parameters of G;
19: **end if**
20: **end for**
21: **end for**
22: modify the pheromone matrix of ACS;
23: $[optimal, S] = ACS(G)$;
24: **end while**
25: **return** S and $optimal$.

7 Experiments and Analysis

Our experiments have two parts: one is to test the performance of the pheromone modification strategies in the dynamic ACS, and the other is to evaluate the performance of the ant-inspired negotiation approach. The results in the first part can help us to decide which pheromone modification strategy will be used in the negotiation-based approach. In our experiments, a trial testing method is adopted to determine most suitable values for all parameters of ACS, considering other researchers' earlier experiments. Finally, the parameters of ACS were $\alpha = 1$, $\beta = 2$, $q_0 = 0.9$, $\rho = 0.1$, $\xi = 0.1$, and $noa = 10$. To evaluate the dynamic ACS and the negotiation approach, we also implement a MIP approach to be applied to the same problem and compare with our approaches. We used the open source integer programming system $lpsolve$ version 5.5 [37] for MIP. All the experiments are conducted on computers with Inter Core i5 2500 CPU (3.3GHz and 8 GB RAM).

7.1 Performance of Different Pheromone Modification Strategies

For the purpose of our evaluation, we tested the dynamic ACS with two different case studies: one with nine abstract services as shown in Fig. 2, the other with thirty abstract services. The second case study was created by either placing a candidate set into Fig. 2 or adding another composition structure as substructure. For each case, 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 1, 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 algorithm, 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 randomly. This procedure continues until the number of replacing services is k.

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 types of 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.

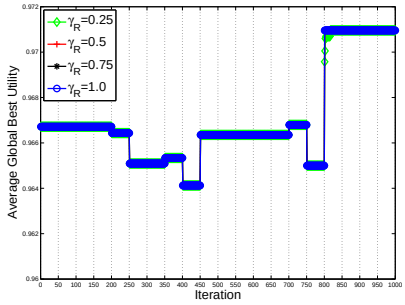
7.1.1 Results of Case Study One

A detailed view of the optimization behavior for each individual strategy and its strategy-specific parameter is shown in Fig. 7. The figure is for the case of $(k, t) = (1, 50)$, small changes occur frequently. After each change occurs, we see a hop of the utility of the best solution in the figure. For example, after $50th$ iteration a change occurs for the first time and after $100th$ iteration a change occurs for the second time. Then the ants need to find a solution for the new scenario between the $51st$ iteration and $100th$ iteration and let fi be the iteration where the best utility for the new scenario presents the first time. The smaller value of $fi - 50$ suggests a quick recovery from the change occurred in $51st$ iteration. From Fig. 7, we observed that the best utility is the same after changes occurred in the interval $[1, 200]$, $[251, 350]$, $[451, 700]$, and $[801, 1000]$. That is to say, the old feasible solution is the same as the new feasible solution after changes occurred in these intervals. For the other combination of k and t, the optimization behavior for each individual strategy is similar with those in Fig. 7.

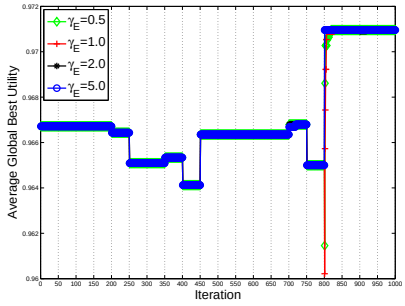
Besides the optimization behavior of each strategy, we also recorded the loss in quality of the best solution compared with the solution found by the MIP approach. As ACS is sub-optimal, we have evaluated the quality of the solution obtained by ACS through comparing with the optimal solution obtained by the MIP approach. The global best utility of the solution obtained by the MIP approach 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 of the best solution, which is given by (14).

$$loss = \frac{U_{global} - U_{acs}}{U_{global}} * 100\% \quad (14)$$

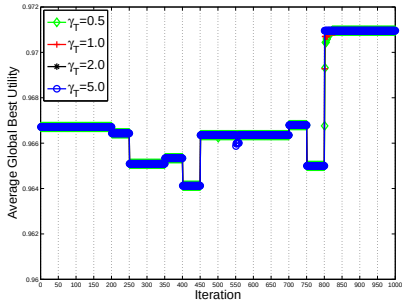
Fig. 8 gives a more detailed view of the performance of each strategy and its different strategy-specific



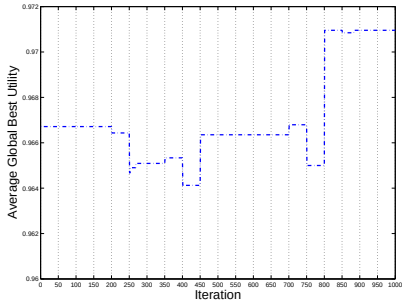
(a) R-strategy with all parameters



(b) η-strategy with all parameters



(c) τ-strategy with all parameters



(d) G-strategy

Fig. 7: The optimization behavior of each strategy in case study one

parameters for the case of $(k, t) = (1, 50)$. The upper row shows the loss of the quality for R-strategy with $\gamma_R \in \{0.25, 0.5, 0.75, 1.0\}$ from left to right, the middle two rows show the loss of the quality for η-strategy and τ-strategy with $\gamma_E, \gamma_T \in \{0.5, 1.0, 2.0, 5.0\}$ from left to right, and the whole lower row show the loss

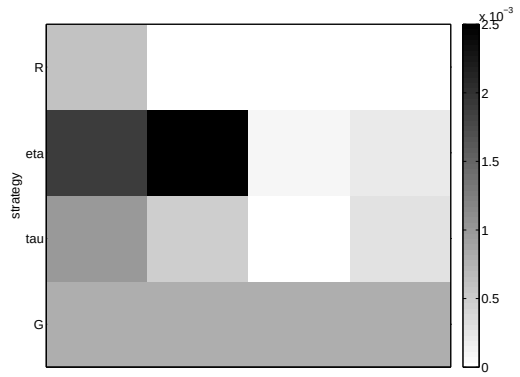


Fig. 8: The loss in quality of the best found solution of all strategies for $(k, t) = (1, 50)$ in case study one

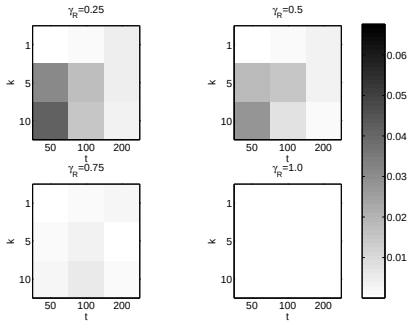
of the quality for G-strategy since there is no strategy-specific parameter in it. The smaller the loss's value, the brighter the grid will be. Judging from the gray scale of the grid, the best utility is the R-strategy with $\gamma_R \in \{0.5, 0.75, 1.0\}$ and τ-strategy with $\gamma_T = 2.0$. The η-strategy with $\gamma_E = 2.0$ is also able to achieve good solutions. The G-strategy is better than the η-strategy with $\gamma_E \in \{0.5, 1.0\}$ and τ-strategy with $\gamma_T = 0.5$.

Fig. 9 shows 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 $\gamma_R = 1.0$, τ-strategy with $\gamma_T = 5.0$, and the G-strategy. The η-strategy with $\gamma_E = 5.0$ provides good solutions when $k = 5$ and $t \in \{100, 200\}$. The loss in G-strategy is almost zero because when the number of abstract services is small, the ants can find the best utility in the first iteration. Although no old pheromone information is preserved, the ants can find new solution quickly. This was illustrated by the experimental results of our earlier study [15].

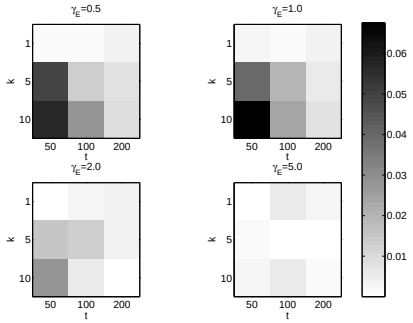
7.1.2 Results of Case Study Two

For case study two, the optimization behavior of each strategy for the case of $(k, t) = (1, 50)$, is shown in Fig. 10. From the figure, we observe that $\gamma_R = 1.0$ in R-strategy, $\gamma_E = 5.0$ in η-strategy, and $\gamma_T = 5.0$ in τ-strategy give better results than other strategy-specific parameters for the individual strategies in general. Compared with the optimization behavior given in Fig. 7, the average global utility at each interval gradually increase. That is because in case study one the ants can find the best utility in just a 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.

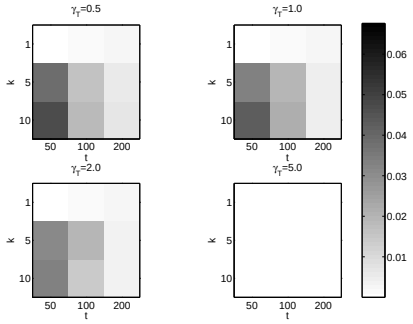
Fig. 11 shows the performance of each strategy and its different strategy-specific parameters for all the cases of the combination of k and t. From the figure, we observe that R-strategy with $\gamma_R = 0.5$



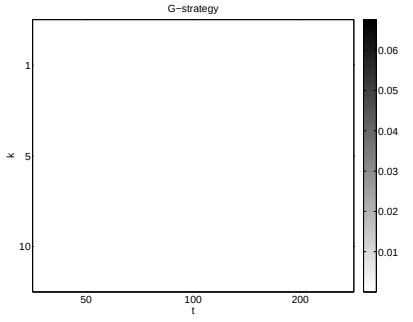
(a) R-strategy



(b) η-strategy



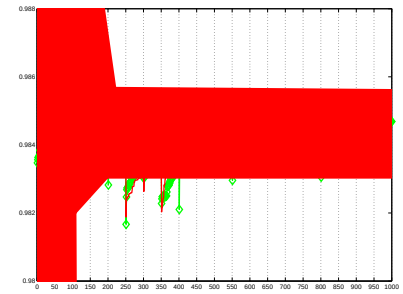
(c) τ-strategy



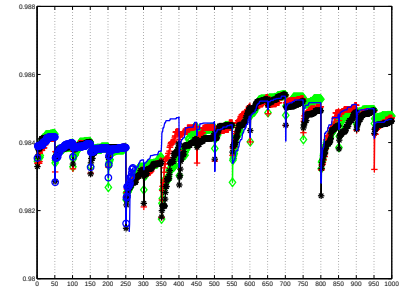
(d) G-strategy

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

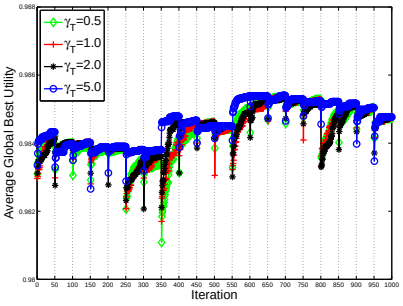
and η-strategy with $\gamma_E = 2.0$ give better solutions when $(k, t) = (1, 200)$, τ-strategy with $\gamma_T = 2.0$ gives better solutions when $(k, t) = (5, 200)$. The G-strategy gives better solutions than $\gamma_E = 5.0$ in η-strategy when $k \in \{5, 10\}$ and $t = 50$. Judging from the



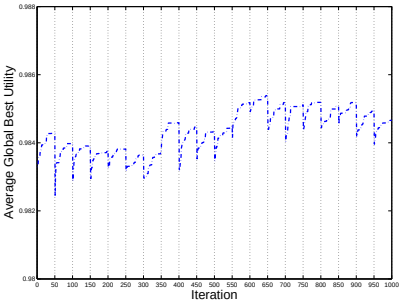
(a) R-strategy with all parameters



(b) η-strategy with all parameters

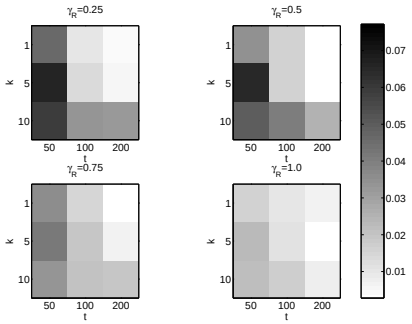


(c) τ-strategy with all parameters

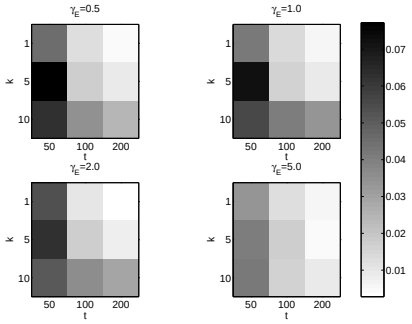


(d) G-strategy

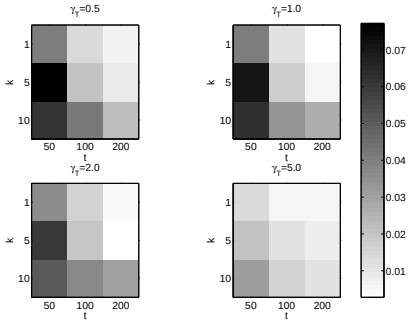
Fig. 10: The optimization behavior of each strategy in case study two



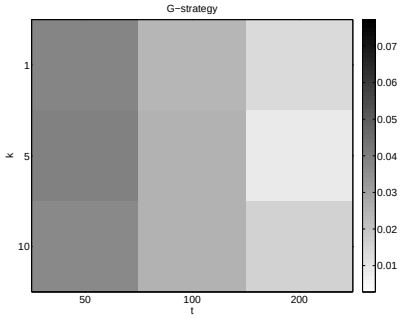
(a) R-strategy



(b) η-strategy



(c) τ-strategy



(d) G-strategy

Fig. 11: The loss in quality of the best found solution of each strategy with different values of strategy-specific parameters, k, and t in case study two

"average darkness", R-strategy with $\gamma_R = 1.0$ gives better solutions when $k \in \{5, 10\}$, and τ-strategy with $\gamma_T = 5.0$ gives better solutions when $k = 1$.

The results of the two case studies indicate that $\gamma_R = 1.0$ in R-strategy, $\gamma_E = 5.0$ in η-strategy, and $\gamma_T = 5.0$ in τ-strategy give better results than other strategy-specific parameters for the individual strategies. The G-strategy could give better results when the number of abstract services is very small, in our case is 10. When the number of abstract services becomes bigger, the R-strategy could give better results in the

case of many changes, and the τ-strategy could give better results in the case of a small number of changes.

7.2 Performance of the Negotiation Approach

The aim of this evaluation is to analyze the performance of the proposed ant-inspired negotiation approach: 1) observing the effectiveness of our approach, measured by the success rate of finding a feasible solution; 2) investigating the efficiency of our approach, measured by the number of negotiation rounds and the computation time in each negotiation round; and 3) studying the effect of the problem size on the performance of our approach.

7.2.1 Test Case Generation

The performance of our approach is affiliated to the size of the data-intensive service provision problem. The size of the problem depends on the number of abstract services used in the composite service, the number of concrete services for each abstract service, and the number of data sets required by each abstract service. For the purpose of our evaluation, we considered different scenarios, where a composite service is composed of n abstract services, and n varies in our experiments between 10 and 100, in increments of 10. There are m concrete services for each abstract service, and m varies in our experiments between 10 and 100, in increments of 10. Each abstract service requires a set of d data sets, and d varies in our experiments between 1 and 10, in increments of 1.

The optimization goal of our experiments is to maximize the overall utility, which was specified in our earlier work [18]. The severe quality constraints were generated according to the method described in [5]. A scenario generation system is designed to generate the scenario for experiments. The system first determines a basic scenario, which includes sequence, conditional and parallel structures. With this basic scenario, other scenarios are generated by either placing an abstract service into it or adding another composition structure as substructure. This procedure continues until the scenario has the predefined number of abstract services. Our simulation scenarios are actually more general and complex than the AMS itself. For each scenario, the price of a data set, the network bandwidth (Mbps) between each data server and the service platform, the storage media speed (Mbps), the size (MB) of a data set and the number of data requests in the waiting queue were randomly generated from the following intervals: [1,100], [1,100], [1,100], [1000,10000] and [1,10]. Each time the number of abstract services, the number of service providers, or the number of data sets was changed, 50 instances of experiments were run and the average results were reported.

7.2.2 Results Analysis

The experimental results of all scenarios are grouped into three test sets. In the first set, n varies from 10 to 100, while $m = 10$ and $d = 5$. In the second set, m varies from 10 and 100, while $n = 10$ and $d = 5$. In the third set, d varies from 1 to 10, while $n = 10$ and $m = 10$.

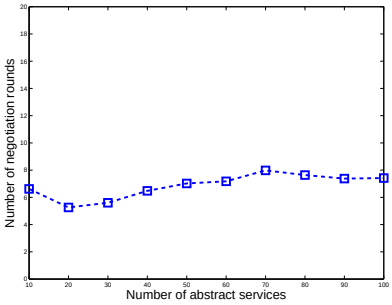
To evaluate the effectiveness of our approach, we compute the success rate of our approach. The success rate is the percentage of instances where a solution could be found. The success rate of the first test set is 92%, 100%, 100%, 98%, 98%, 100%, 94%, 98%, 100%, and 98%. The success rate of the second test set is 92%, 100%, 98%, 100%, 100%, 100%, 98%, 100%, 100%, and 96%. The success rate of the third test set is 92%, 100%, 98%, 98%, 92%, 94%, 98%, 94%, 90%, and 96%. For the MIP approach, the success rate is zero because it cannot work when the QoS constraints are very severe. The experimental results show that our negotiation approach maintains a higher success rate. Specifically, the average success rate of our approach is 97.8%, 98.4%, 95.20% in the first, second, and third test set.

The number of negotiation rounds is a relevant efficiency factor of the negotiation approach. A large value in the negotiation round implies that the problem has to be solved via a large number of iterations in the whole negotiation process. Fig. 12 shows the negotiation rounds of the three test sets. The figures indicate that the number of negotiation rounds increases when the number of service providers increases. When the number of abstract services or the number of data sets increases, the number of negotiation rounds, on the other hand does not exact increase. On average, it took approximately seven rounds to find a feasible solution in the first and third test sets.

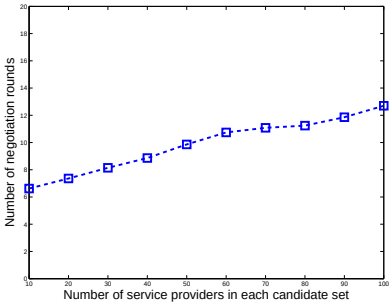
Besides the number of negotiation rounds, the time consumption for each round is also a very important factor with respect to the efficiency. Fig. 13 presents the average time consumption per negotiation round of the three sets. The results indicate that the time consumption per round increases when the number of abstract services, the number of service providers, or the number of data sets increases. As illustrated, the time consumption per round is linear rather than exponential.

8 Conclusion

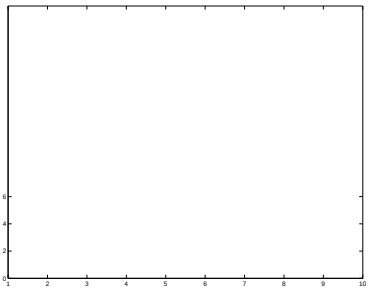
The provision of data-intensive service faces new challenges with the rapid proliferation of services and the development of cloud computing. The scope, number, and complexity of data-intensive services are all set to soar in the future. This paper proposed an ant-inspired negotiation approach for the data-intensive service provision. The two-stage negotiation procedures provided effective and efficient service



(a) Varying number of abstract services

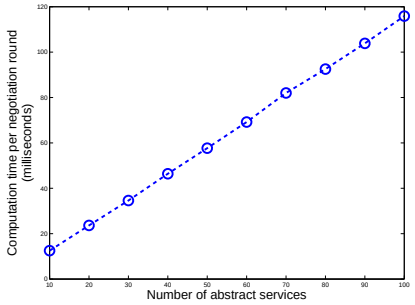


(b) Varying number of service providers

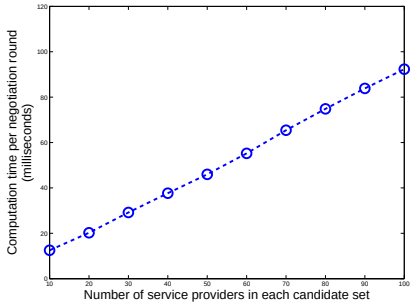


(c) Varying number of data sets

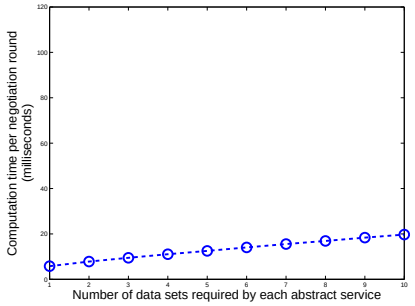
Fig. 12: Number of negotiation rounds



(a) Varying number of abstract services



(b) Varying number of service providers



(c) Varying number of data sets

Fig. 13: Computation time per negotiation round

selection for service composers. A multi-phase, multi-party negotiation protocol was also presented, where the ant colony system was applied for selecting the services. In order to adapt the ant colony system to handle the dynamic scenarios during negotiations, we also discussed several strategies for modifying the pheromone information. The experimental results indicated that our ant-inspired negotiation approach could facilitate the data-intensive applications in the AMS experiment. The proposed negotiation approach present features such as autonomy, scalability, and adaptability.

In future work, we will further investigate the economic or technical strategies that service providers and data providers can deploy to improve offers. For example, the service provider can subscribe data sets and move some lightweight data sets to the service platform, or the data provider can relocate data sets in order to decrease the response time of moving data sets. These strategies can improve the success rate in finding feasible solutions and decrease the total cost of the data-intensive service provision.

REFERENCES

[1] J. Cao, J. Wang, S. Zhang, and M. Li, "A Multi-Agent Negotiation Based Service Composition Method for On-Demand Service," in *Proceedings of IEEE International Conference on Services Computing*. Washington, DC, USA: IEEE Computer Society, 2005, pp. 329–332.

[2] C. Di Napoli, "Software Agent Negotiation for Service Composition," in *Agent and Multi-Agent Systems: Technologies and Applications*, ser. Lecture Notes in Computer Science, N. Nguyen, A. Grzech, R. Howlett, and L. Jain, Eds. Berlin, Heidelberg: Springer-Verlag, 2007, vol. 4496, pp. 456–465.

[3] M. Navarro, E. Val, M. Rebollo, and V. Julián, "Agent Negotiation Protocols in Time-Bounded Service Composition," in *Intelligent Data Engineering and Automated Learning*, ser. Lecture Notes in Computer Science, E. Corchado and H. Yin, Eds. Berlin, Heidelberg: Springer-Verlag, 2009, vol. 5788, pp. 527–534.

[4] F. H. Zulkernine and P. Martin, "An Adaptive and Intelligent SLA Negotiation System for Web Services," *IEEE Transactions on Services Computing*, vol. 4, no. 1, pp. 31–43, 2011.

[5] Q. He, J. Yan, H. Jin, and Y. Yang, "Quality-Aware Service Selection for Service-Based Systems Based on Iterative Multi-Attribute Combinatorial Auction," *IEEE Transactions on Software Engineering*, vol. 40, no. 2, pp. 192–215, 2014.

[6] C. L. Philip Chen and C. Y. Zhang, "Data-Intensive Applications, Challenges, Techniques and Technologies: A Survey on Big Data," *Information Sciences*, vol. 275, pp. 314–347, 2014.

[7] 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.

[8] D. Ardagna and B. Pernici, "Adaptive Service Composition in Flexible Processes," *IEEE Transactions on Software Engineering*, vol. 33, no. 6, pp. 369–384, 2007.

[9] 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.

[10] P. N. Bless, D. Klabjan, and S. Y. Chang, "Heuristics for Automated Knowledge Source Integration and Service Composition," *Computers and Operations Research*, vol. 35, no. 4, 2008.

[11] 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.

[12] W. Zhang, C. K. Chang, T. Feng, and H. Y. Jiang, "QoS-Based Dynamic Web Service Composition with Ant Colony Optimization," in *Proceedings of the 34th Annual IEEE International Computer Software and Applications Conference (COMPSAC '10)*. Washington, DC, USA: IEEE Computer Society, 2010, pp. 493–502.

[13] G. Canfora, M. Penta, R. Espositio, 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.

[14] 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.

[15] L. Wang and J. Shen, "A Critical Systematic Review of Service Concretization Based on Bio-Inspired Approaches," 2014, http://ro.uow.edu.au/eispapers/1903.

[16] S. Balasubramaniam, D. Botvich, R. Carroll, J. Mineraud, T. Nakano, T. Suda, and W. Donnelly, "Biologically Inspired Future Service Environment," *Computer Networks*, vol. 55, no. 15, pp. 3423–3440, 2011.

[17] L. Wang and J. Shen, "Towards Bio-Inspired Cost Minimization for Data-Intensive Service Provision," in *IEEE First International Conference on Services Economics*. Washington, DC, USA: IEEE Computer Society, 2012, pp. 16–23.

[18] 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.

[19] 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.

[20] 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 *Proceedings of the IEEE 2nd International Congress on Big Data*. Washington, DC, USA: IEEE Computer Society, 2013, pp. 254–261.

[21] 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.

[22] L. Wang, J. Shen, J. Luo, and F. Dong, "An Improved Genetic Algorithm for Cost-Effective Data-Intensive Service Composition," in *The 9th International Conference on Semantics, Knowledge & Grids*. Washington, DC, USA: IEEE Computer Society, 2013, pp. 105–112.

[23] Y. Li and C. Lin, "QoS-Aware Service Composition for Workflow-Based Data-Intensive Applications," in *Proceedings of IEEE International Conference on Web Services*. Washington, DC, USA: IEEE Computer Society, 2011, pp. 452–459.

[24] N. R. Jennings, P. Faratin, A. R. Lomuscio, S. Parsons, M. J. Wooldridge, and C. Sierra, "Automated Negotiation: Prospects, Methods and Challenges," *Group Decision and Negotiation*, vol. 10, no. 2, pp. 199–215, 2001.

[25] J. Yan, R. Kowalczyk, J. Lin, M. B. Chhetri, S. K. Goh, and J. Zhang, "Autonomous Service Level Agreement Negotiation for Service Composition Provision," *Future Generation Computer Systems*, vol. 23, no. 6, pp. 748–759, 2007.

[26] F. Siala, S. Lajmi, and K. Ghédira, "Multi-Agent Selection of Multiple Composite Web Services Based on CBR Method and Driven by QoS," in *Proceedings of the 13th International Conference on Information Integration and Web-Based Applications and Services (iiWAS'11)*. New York, NY, USA: ACM, 2011, pp. 90–97.

[27] F. Siala and K. Ghédira, "How to Select Dynamically a QoS-Driven Composite Web Service by a Multi-Agent System Using CBR Method," *International Journal of Wireless and Mobile Computing*, vol. 7, no. 4, pp. 327–347, 2014.

[28] M. Dorigo and T. Stutzle, *Ant Colony Optimization*. Cambridge, MA, USA: MIT Press, 2004.

[29] 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.

[30] 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.

[31] FIPA, "Iterated Contract Net Interaction Protocol Specification," 2002, http://www.fipa.org/specs/fipa00030/SC00030H.pdf.

[32] Z. Alibhai, "What is Contract Net Protocol," 2003, http://www2.ensc.sfu.ca/research/iDEA/courses/files/ContractNetProtocol1.pdf.

[33] K. Hashmi, A. Alhosban, Z. Malik, and B. Medjahed, "WebNeg: A Genetic Algorithm Based Approach for Service Negotiation," in *2011 IEEE International Conference on Web Services (ICWS)*. Washington, DC, USA: IEEE Computer Society, 2011, pp. 105–112.

[34] P. Faratin, C. Sierra, and N. R. Jennings, "Negotiation Decision Functions for Autonomous Agents," *Robotics and Autonomous Systems*, vol. 24, no. 3-4, pp. 159–182, 1998.

[35] M. B. Chhetri, J. Lin, S. Goh, J. Zhang, R. Kowalczyk, and J. Yan, "A Coordinated Architecture for the Agent-Based Service Level Agreement Negotiation of Web Service Composition," in *Proceedings of the Australian Software Engineering Conference (ASWEC '06)*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 90–99.

[36] H. Li, S. Y. W. Su, and H. Lam, "On Automated e-Business Negotiations: Goal, Policy, Strategy, and Plans of Decision and Action," *Journal of Organizational Computing and Electronic Commerce*, vol. 13, no. 1, pp. 1–29, 2006.

[37] M. Berkelaar, K. Eikland, and P. Notebaert, "lpsolve: Open