University of Wollongong

Research Online

Faculty of Engineering and Information
Sciences - Papers: Part A

Faculty of Engineering and Information
Sciences

1-1-2013

Cost and time aware ant colony algorithm for data replica in alpha magnetic spectrometer experiment

Lijuan Wang

University of Wollongong, lw840@uowmail.edu.au

Junzhou Luo

Southeast University

Jun Shen

University of Wollongong, jshen@uow.edu.au

Fang Dong

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

Cost and time aware ant colony algorithm for data replica in alpha magnetic spectrometer experiment

Abstract

Huge collections of data have been created in recent years. Cloud computing provides a way to enable massive amounts of data to work together as data-intensive services. Considering Big Data and the cloud together, which is a practical and economical way to deal with Big Data, will accelerate the availability and acceptability of analysis of the data. Providing an efficient mechanism for optimized data-intensive services will become critical to meet the expected growth in demand. Because the competition is an extremely important factor in the marketplace, the cost model for data-intensive service provision is the key to provide a sustainable service market. As data play the dominant role in execution of data-intensive service composition, the cost and access response time of data sets influence the quality of the service that requires the data sets. In this paper, a data replica selection optimization algorithm based on an ant colony system is proposed. The performance of the data replica selection algorithm is evaluated by simulations. The background application of the work is the Alpha Magnetic Spectrometer experiment, which involves large amounts of data being transferred, organized and stored. It is critical and challenging to be cost and time aware to manage the data and services in this intensive research environment.

Keywords

experiment, aware, ant, cost, colony, time, algorithm, data, replica, alpha, magnetic, spectrometer

Disciplines

Engineering | Science and Technology Studies

Publication Details

Wang, L., Luo, J., Shen, J. & Dong, F. (2013). Cost and time aware ant colony algorithm for data replica in alpha magnetic spectrometer experiment. IEEE 2nd International Congress on Big Data (pp. 254-261). United States: IEEE.

Cost and time aware ant colony algorithm for data replica in Alpha Magnetic Spectrometer experiment

Lijuan Wang¹, Junzhou Luo², Jun Shen¹, Fang Dong²

¹School of Information Systems and Technology
University of Wollongong, NSW, Australia

²School of Computer Science and Engineering
Southeast University, Nanjing, P. R. China

E-mail: lw840@uowmail.edu.au, jluo@seu.edu.cn, jshen@uow.edu.au, fdong@seu.edu.cn

Abstract—Huge collections of data have been created in recent years. Cloud computing provides a way to enable massive amounts of data to work together as data-intensive services. Considering Big Data and the cloud together, which is a practical and economical way to deal with Big Data, will accelerate the availability and acceptability of analysis of the data. Providing an efficient mechanism for optimized data-intensive services will become critical to meet the expected growth in demand. Because the competition is an extremely important factor in the marketplace, the cost model for data-intensive service provision is the key to provide a sustainable service market. As data play the dominant role in execution of data-intensive service composition, the cost and access response time of data sets influence the quality of the service that requires the data sets. In this paper, a data replica selection optimization algorithm based on an ant colony system is proposed. The performance of the data replica selection algorithm is evaluated by simulations. The background application of the work is the Alpha Magnetic Spectrometer experiment, which involves large amounts of data being transferred, organized and stored. It is critical and challenging to be cost and time aware to manage the data and services in this intensive research environment.

Keywords—service provision, data-intensive, ant colony optimization, cloud computing, Big Data, QoS

I. INTRODUCTION

In recent years, huge collections of data have been created by the advances in technology areas such as digital sensors, communications, computation, and storage. Scientists and computer engineers have coined a new term for the phenomenon: “Big Data”. Big Data allows us to discover much more about the world, not only using structured internal data but also using unstructured data from external sources. Cloud computing has become a viable, mainstream solution for data processing, storage, and distribution. It provides unlimited resources on demand. Considering Big Data and the cloud together, which is a practical and economical way to deal with Big Data, will accelerate the availability and acceptability of analysis of the data. To put Big Data to work, increasing numbers of companies are starting to use the cloud to publish Big Data as a data service. Many data services in the area of Big Data analytics have now become available. Cloud infrastructure and platforms will play an important role in accessing, processing, and analyzing massive amounts of data, all of which are concerned with huge cost and energy consumption to maintain the data centers.

Like business sectors, where Big Data offerings are cross-organizational, Big Data projects are normally multidisciplinary in the scientific area, and on a multinational scale. The background of this work is the Alpha Magnetic Spectrometer (AMS) experiment which uses cloud computing to process huge amounts of data. The AMS experiment is a large-scale international collaborative project, and the only large physics experiment on the International Space Station. The purpose of the AMS experiment is to study the universe and its origin by searching for antimatter and dark matter while performing precision measurements of cosmic rays composition and flux. The AMS lab is based at CERN in Switzerland. The key technology for accessing the data remotely collected from AMS relies on data services based on the cloud computing. This is actually supported by IBM Cloud Computing Center located at Southeast University in China. Typically, at our center, we receive 200G bytes data from AMS and generate 700G bytes data after processing them, on each single day. To retrieve and access the data, we use dedicated 40Gbps InfiniBand network connectors between CERN and Southeast University, and our Cloud Computing Center has IBM servers with 3500 core and 500TB storage, which is shown in Fig. 1.

To solve a complex scientific problem such as in the AMS experiment, scientists need to combine data from various sources. It is necessary to design a workflow of various data-intensive services and get a composite data-intensive service when using Web service technologies. The cost and response time of each service in the data-intensive service composition are critical for the composite service. The data-intensive service composition has the following challenges. First, large number of data sets and increasing functionally equivalent services make the composition complex. Second, the size and

Fig. 1: IBM Cloud Computing Center in Southeast University

the number of distributed data sets make the communication and storage costs increase, which effect the performance of the whole application process. Third, the cost of transferring data to and from service endpoints increases as the number of data sets increase. Fourth, the dynamic nature of cloud computing and data replication need a dynamic and adaptive mechanism to regulate the interaction between users and providers.

In [19], it was pointed out that a service composer would need a cost-control mechanism to limit spending on the constituent services, that is to say, minimizing the cost of service provision. Similarly, service providers will also need a cost-control mechanism to limit spending on data sets, if they are to maintain a competitive position and win contracts with the service composer. Each service requests data sets from the storage resources (or data servers). Each of these data sets may be replicated at several locations (typically three) that are connected to each other and to the service endpoints through networks of varying capability [17], [27], [34]. Users can access data remotely from a data server or store a copy of data locally for future use. As data play the dominant role in execution of data-intensive service composition, the movement of mass data influences the performance of the whole process. Especially, the access cost of each data replica on one data server is different from that on other data servers [30], and the cost of service relates to the amount of data transferred. Thus, it is necessary to present a data replica selection mechanism for service providers to lower the costs of services and improve the qualities of services.

In this paper, we address the challenges listed above for data-intensive service composition by making the following two contributions. First, we give a cost and time aware model for data replica. The model has presented the access response time and cost of each data replica. Second, we model the data replica selection problem as a multi-objective optimization problem using an ant colony system algorithm. The ants select data sets from multiple data replicas based on the cost and response time of transferring data from data servers to service endpoints, so that the total cost and response time of a service is minimized.

The remainder of this paper is organized as follows. The next Section reviews related work. Section III introduces the foundation problem and data cost model. Section IV details the data replica selection. Section V investigates how an ant colony system algorithm could be used to solve data replica selection problem. Section VI shows the experiments and analysis. Finally, Section VII concludes this paper.

II. RELATED WORK

Data replication is considered to be an important technique used in cloud computing to speed up data access, reduce bandwidth consumption, execution cost and users waiting time, and increase data availability [14]. Data is replicated across large geographic distances in the cloud [1]. Given the dynamic distribution of data and data requests from users, the “best” replica site for users to access data is determined based on some criteria by the replica selection process. The user response time is the most relevant metric in the existing works on replica selection. As this metric cannot be computed in advance, some other factors such as network and server performance are adopted to estimate the response time.

The literature has provided two types of approaches for replica selection. The static replica selection approaches select the nearest replica to the user according to some static metric such as the geographical distance in miles, topological distance in number of hops, and HTTP request latency [9], [11], [12], [25]. The experimental results in [24], [36] showed the static metrics ignored the network dynamic conditions so they were not sufficient predictors for the expected response time for user requests. The dynamic replica selection approaches have emerged to improve the estimation of the expected user response time based on network factors such as round-trip time, network bandwidth, and server request latency [6], [10], [13], [20], [21], [37].

Quite a few replica selection strategies are proposed for data Grid. In [7], we proposed an effective data aggregation based adaptive long term resource load point prediction mechanism, where a data aggregation concept is introduced to reduce the number of prediction step. The authors of [21] used a neural network algorithm to predict the transfer time for different sites that hold replicas and then proposed a k-nearest neighbor rule to select the best replica. Their goal is to minimize the access latency. Since the status of data replica changes dynamically, the neural network algorithm does not always give the right decision. Also, the misclassification in k-nearest neighbor rule will increase when large files are transferred and the rule needs to save all previous requests which needs time to search [3]. The paper [4] used a reverse Vickrey auction to select the cheapest replica of a file. The cost of a replica is proportional to the combination of access time and waiting time in the queue. The economic model using auction protocol to select the cheapest data replica performs long term optimization, however, it is not efficient for single data request.

A few studies propose replica selection strategies based on bio-inspired algorithms [16], [18], [28]. Bio-inspired algorithms offer many advantages for dealing with data-intensive service provision problems [26], [32]. Bio-inspired optimization algorithms have been proposed to solve the service provision problem, because of the simplicity of the algorithms and the rapid convergence to optimal or near-optimal solutions [15], [31], [33]. Biological entities can learn from their environment. They can sense the surrounding conditions and adaptively invoke behaviors suited to the conditions. Biological inspired systems are typically made of a population of simple agents, which try to build the feasible solution to apply the stochastic decision policy repeatedly. They are decentralized and self-organized systems. In order to deal with the dynamic changes of data replicas and network conditions in cloud computing, as well as the constraints of different users and the flexibility of the selection criteria, the ant colony optimization algorithm will be used to solve data replica selection problem in this paper. To the best of our knowledge, this is the first work that shows how to use ant colony optimization algorithm to solve data replica selection problem.

III. BACKGROUND

A. Foundation problem

In general, data-intensive service composition will be supported cooperatively by service composers, service providers,

Fig. 2: Service selection and data replica selection graph

and data providers. The service composer seeks optimal strategies to select elementary services provided by multiple service providers. From the service composer's point of view, it is important to be able to assess the value of the needed services and how much it wants to pay for them to satisfy its users' requirements as well as to minimize its cost. From the service provider's perspective, it is important to be able to analyze its competitive position and improve its offers if it is to win contracts with the service composer. The service provider requests the data from the data provider. The costs of the data affect the total cost of services. Therefore the costs of service and data have a crucial impact on the cost of service composition.

A data-intensive service composition environment can be considered to consist of a set of z data servers, $D = \{d_1, d_2, \dots, d_z\}$. Suppose that a composite service CS is composed of a set of n abstract services, which is denoted by $AS = \{AS_1, AS_2, \dots, AS_n\}$ (Note: Since a composite service is similar to a workflow [5], tasks and abstract services are used interchangeably). It is assumed that there are m concrete services, $cs_i = \{cs_{i,1}, cs_{i,2}, \dots, cs_{i,m}\}$, mapping to each abstract service AS_i . Each concrete service $cs_{i,j}$ is associated with a QoS vector $q_{ij} = [q_{ij}^1, q_{ij}^2, \dots, q_{ij}^r]$ with r QoS parameters. Each abstract service, $AS_i \in AS$, requires a set of k data sets, denoted by DT^i , that are distributed on a subset of D . Each data set $dt \in DT^i$ has l data replicas. Specifically, for a data set $dt \in DT^i$, $D^{dt} \subseteq D$ is the set of data servers (each denoted by d_{dt}) on which dt is replicated and from which it is available. Also, a data server can hold multiple data sets at a time. Fig. 2 describes the service selection and data replica selection.

Consider that a data-intensive service $cs_{i,j}$ has been chosen to replace AS_i , the service endpoint is connected by links of different bandwidths with all the data servers. The cost for the task AS_i includes three parts: the data access cost, the data transfer cost, and the service related cost. The data access cost is the sum of the price of all data sets. The price of a data set is the fee that a data user has to pay to the data provider for the data usage. The data transfer cost are proportional to the transfer time, which depends on the available network bandwidth between data server and service endpoint. The service related cost which mainly includes the cost to provision the service and the cost to process data sets.

Thus, selecting the reasonable data replicas can improve the response time of the service and also decrease the cost of the service.

B. Data cost model

The data providers use different pricing models to supply data sets, such as the usage-based pricing model, the package-based pricing model, the flat fee subscription-based pricing model, and the combination-based pricing model.

In the usage-based pricing model (UB), users need to pay data sets for respective usage. This is also called the short-term option. The base price for an access of data set dt is denoted by p_{dt} , the total cost of u_{dt} usages of dt is given by (1).

$$cost^{UB}(dt, u_{dt}) = p_{dt} * u_{dt} \quad (1)$$

In the package-based pricing model (PB), the data provider offers users a certain amount of data sets for a fixed fee. For example, if users use two data sets frequently and the price of the package of the two data sets is lower than the sum of individual data set, then the user can purchase the package. Assume a user needs a set of pk data sets, the base price for an access of each data set dt^v is denoted by p_{dt^v} ($v \in \{1, 2, \dots, pk\}$). The data provider offers a package of pk data sets for a price of p_{pk} , and $\sum_{v=1}^{pk} p_{dt^v} < p_{pk}$. Thus, the total cost of u_{dt} usages of pk data sets is given by (2).

$$cost^{PB}(dt^{1,2,\dots,pk}, u_{dt}) = p_{pk} * u_{dt} \quad (2)$$

In the flat fee subscription-based pricing model (SB), users need to pay once for a data set and afterwards they can access the data set for a period of time. The cost of access the data set in this period is independent from the number of usages. This is also called the long-term option. In this model, service providers can move the data set closer to service endpoint that requires it. The total cost of data set dt during the period of st is given by (3).

$$cost^{SB}(dt, st) = f_{dt}^{st} \quad (3)$$

where f_{dt}^{st} is the fee of data set dt during the period of st .

In the combination-based pricing model (CB), the service provider pays a flat fee and surcharge depending on the usage of the data set. In this model, the total cost of u_{dt} usages of dt is given by (4).

$$cost^{CB}(dt, u_{dt}) = \begin{cases} f_{dt}, u_{dt} \leq U_{dt} \\ f_{dt} + \alpha_{dt} * (u_{dt} - U_{dt}), u_{dt} > U_{dt} \end{cases} \quad (4)$$

where f_{dt} is the flat fee of dt , U_{dt} represents the maximal amount of usages of dt , α_{dt} denotes the surcharge of each access when the usage of dt extends the threshold U_{dt} .

The service providers (data users) can switch from one pricing model to another pricing model back and forth, according to their demands. If they choose the flat fee subscription-based pricing model or the combination-based pricing model, they can store a copy of the data sets locally. This can reduce the response time and enhance the usage of bandwidth and other QoS attributes.

IV. DATA REPLICA SELECTION

A. Agent based data replica

The selection of data replicas is performed by a set of data replica (DR) agents. DR agents are located on each service endpoint and use an optimization approach for selecting the optimal replica of a data set and a caching function to make informed decisions about local data caching. As the authors in [22] distinguished caching and replication, replication is assumed to be a server side phenomenon that the server decides when and where to create a replica of its data, where caching is defined as a user side phenomenon that the user selects the best replica and caches a copy of the replica at the local machine. A DR agent is invoked by a service that needs to access a data set. If the service endpoint does not hold the data set, the DR agent consults replica location service [23] for replica locations. Then DR agent uses the optimization approach to select the optimal replica of that data set, and it might invoke the caching function. The purpose of the caching function is to create, on the corresponding service endpoint, a copy of the requested data set. A caching function is only invoked if the service broker decides that having a local copy of the data set is economically beneficial, in the situation that the service endpoint has space to store the data set.

B. The data access response time

In order to lower the cost of data-intensive service composition solution, this paper regards the response time and the cost of a data replica as the criteria for the data replica selection process. The data access response time is defined as the time that elapses from when a service requests for a data set until it receives the complete data set. Since the data-intensive service composition problem in this paper is concerned with the time to complete a task, it also takes into account the storage request queue (data server load) and the storage media speed (I/O data transfer rate), which are mentioned in [2], [4], [29].

Thus, the access response time of data set dt , $T_{rt}(dt)$, includes the data transfer time $T_t(dt, d_{dt}, y)$, the storage access latency $T_{sal}(d_{dt})$, and the request waiting time $T_{wt}(d_{dt})$, which can be calculated by (5).

$$\begin{aligned} T_{rt}(dt) &= T_t(dt, d_{dt}, y) + T_{sal}(d_{dt}) + T_{wt}(d_{dt}) \\ T_t(dt, d_{dt}, y) &= \text{size}(dt)/bw(d_{dt}, y) \\ T_{sal}(d_{dt}) &= \text{size}(dt)/sp(d_{dt}) \\ T_{wt}(d_{dt}) &= \sum_{i=1}^{nr} (\text{size}(dt^i)/sp(d_{dt})) \end{aligned} \quad (5)$$

where $\text{size}(dt)$ is the size of data set dt , $bw(d_{dt}, y)$ is the network bandwidth between data server d_{dt} and service endpoint y , $sp(d_{dt})$ is the storage media speed, nr is the number of data requests waiting in the queue prior to the underlying request for dt . The data transfer time $T_t(dt, d_{dt}, y)$ is the time to transfer the data set from the remote site that houses the data replica to the local site which has the service that requested the replica. It depends on the network bandwidth and the size of the data replica. The storage access latency is the delayed time for the storage media to serve the requests and it depends on the size of the data and storage type [2]. Each storage media has many requests at the same time and it serves only one request at a time. The current request needs to wait until all requests prior it in the queue finish.

C. Cost and time aware model for data replica

For task AS_i requires a set of k data sets, denoted by $DT^i = \{dt^1, dt^2, \dots, dt^k\}$, that are distributed on a subset of D . It is assumed that there are l data servers on which each data set $dt^v \in DT^i$ ($v \in \{1, 2, \dots, k\}$) is replicated and from which it is available. For data set dt^v , the price of its replica is denoted by $p^v = \{p_1^v, p_2^v, \dots, p_l^v\}$, the network bandwidth between its replica site and service endpoint is denoted by $b^v = \{b_1^v, b_2^v, \dots, b_l^v\}$, and the data transfer rate of its replica site is denoted by $sp^v = \{sp_1^v, sp_2^v, \dots, sp_l^v\}$. A binary decision variable x_q^v ($q \in \{1, 2, \dots, l\}$) is used to represent only one data server is selected for each data set, where x_q^v is set to 1 if data server d_q^v is selected to access data set dt^v and 0 otherwise. The cost of data sets for AS_i , $Cost(DT^i)$, can be described by (6).

$$\begin{aligned} Cost(DT^i) &= \sum_{v=1}^k Cost(dt^v) \\ Cost(dt_q^v) &= \sum_{q=1}^l x_q^v \left(p_q^v + (\text{size}(dt^v)/b_q^v) * tcost \right) \end{aligned} \quad (6)$$

The data response time, $T_{rt}(DT^i)$, can be described by (7).

$$\begin{aligned} T_{rt}(DT^i) &= \sum_{v=1}^k T_{rt}(dt_q^v) \\ T_{rt}(dt_q^v) &= \sum_{q=1}^l x_q^v \left(\text{size}(dt^v)/b_q^v \right. \\ &\quad \left. + \text{size}(dt^v)/sp_q^v + \sum_{r=1}^{nr} (\text{size}(dt^r)/sp_q^v) \right) \end{aligned} \quad (7)$$

Subject to:

$$\sum_{q=1}^l x_q^v = 1, x_q^v \in \{0, 1\}, v \in \{1, 2, \dots, k\}. \quad (8)$$

For each data replica dt_q^v , the goal of data replica selection process is to minimize its cost and response time. Hence, in this paper the data replica selection is a multi-objective optimization problem. In order to evaluate the quality of a data replica, a utility function is used and then the problem is transformed into a single objective optimization problem. This paper adopts the simple additive weighting (SAW) approach in the multiple criteria decision making (MCDM) [35] technique for the utility function. The utility computation includes two phases: the scaling phase and the weighting phase. The scaling phase is used to normalize all data replica selection criteria to the same scale, independent of their units and ranges. The weighting phase is used to compute the overall utility for each data replica by using weights depending on users' priorities and preferences.

The utility of a data replica dt_q^v is computed according to (9).

$$\begin{aligned} U(dt_q^v) &= \frac{C^{max} - Cost(dt_q^v)}{C^{max} - C^{min}} * \lambda_1 \\ &\quad + \frac{T^{max} - T_{rt}(dt_q^v)}{T^{max} - T^{min}} * \lambda_2 \end{aligned} \quad (9)$$

where $\lambda_1, \lambda_2 \in [0, 1]$, and $\lambda_1 + \lambda_2 = 1$. λ_1 and λ_2 represent weights of cost and response time of data replica with values normally provided by the users based on their own preferences. C^{max} and C^{min} are the maximum and minimum value of the cost of all replicas of data set dt^v , and T^{max} and T^{min} are the maximum and minimum value of the access response time of all replicas of data set dt^v .

The utility of a set of k data sets, DT^i , is given by (10).

$$U(DT^i) = \frac{C^{MAX} - Cost(DT^i)}{C^{MAX} - C^{MIN}} * \lambda_1 + \frac{T^{MAX} - T_{rt}(DT^i)}{T^{MAX} - T^{MIN}} * \lambda_2 \quad (10)$$

where $C^{MAX} = \sum_{v=1}^k C^{max}$, $C^{MIN} = \sum_{v=1}^k C^{min}$, $T^{MAX} = \sum_{v=1}^k T^{max}$, and $T^{MIN} = \sum_{v=1}^k T^{min}$. Thus, for each task AS_i , the optimization problem is to find a set of data servers to access each data set such that $U(DT^i)$ is maximized.

V. DYNAMIC DATA REPLICA SELECTION BASED ON AN ANT COLONY SYSTEM ALGORITHM

The field of ‘ant colony algorithms’ studies models derived from the behavior of real ants and it is widely used for combinatorial optimization problems. Ant colony algorithms are developed as heuristic methods to identify efficient selections and have been applied to identify optimal solutions for service composition problems. The features of ant colony algorithms include positive feedback and local heuristics. An ant colony optimization (ACO) algorithm iteratively performs a loop containing two basic procedures. The first is how the ants construct solutions to the problem, and the second is how to update the pheromone trails. Using ACO algorithms to solve combinatorial optimization problems, requires a representation of the problem and the definition of the meaning of pheromone trails, as well as the heuristic information.

The ant colony system (ACS) is an algorithm inspired by the ant system (AS) but it differs from AS in three main aspects [8]. First, the state transition rule provides a direct way to balance between exploration of new replicas and exploitation of *a priori* and accumulated knowledge about the problem. Second, the global updating rule is applied only to replicas which belong to the best ant selection. Third, a local pheromone updating rule is applied while ants construct a solution.

A. State transition rule

When a DR agent is invoked by a service, all ants are initially positioned on the service endpoint. After the DR agent gets all the replica location of each data set, the ant moves from the service endpoint and visits each data replica and then return to the service endpoint. When ant k arrives at replica i , it will choose j to move to by applying the rule given by (11).

$$j = \begin{cases} \arg \max_{l \in D_i^k} \{[\tau_l][\eta_l]^\beta\}, & \text{if } q \leq q_0; \\ J, & \text{otherwise.} \end{cases} \quad (11)$$

where q is a random variable uniformly distributed in $[0, 1]$, $q_0 (0 \leq q_0 \leq 1)$ is a parameter, and J is a random variable

selected according to the probability distribution given by (12) (with $\alpha = 1$).

$$p_{ij}^k = \begin{cases} \frac{[\tau_j]^\alpha [\eta_j]^\beta}{\sum_{l \in D_i^k} [\tau_l]^\alpha [\eta_l]^\beta}, & \text{if } j \in D_i^k; \\ 0, & \text{otherwise.} \end{cases} \quad (12)$$

where p_{ij}^k represents the probability with which ant k , currently at replica i , chooses to go to replica j . D_i^k is the set of replicas that ant k has not accessed yet. α is a parameter to control the influence of τ_j , β is a parameter to control the influence of η_j . τ_j is the pheromone density of replica j . Here, $\eta_j = U(dt^j)$ which is computed according to (9) as heuristic information.

B. Global update rule

After all ants arrive at the service endpoint again, a global pheromone updating rule is performed. The pheromone level is updated by applying the global updating rule (13).

$$\tau_j = (1 - \rho)\tau_j + \rho\Delta\tau_j \quad (13)$$

where $\rho (0 < \rho < 1)$ is the pheromone evaporation rate, and

$$\Delta\tau_j = \begin{cases} U, & \text{if } \forall j \in D^{bs}; \\ 0, & \text{otherwise.} \end{cases}$$

where U is the utility of D^{bs} which is computed according to (10). D^{bs} is the best set of data servers found since the start of the algorithm. This formula indicates the pheromone trail update, both the evaporation and the new pheromone deposit, only apply to the replicas of D^{bs} . There are two types of global updating rule. One is ‘iteration-best’, the best selection in the current iteration of the trial, and the other is ‘global-best’, the best selection from the beginning of the trial. Experiments have shown that the global-best is slightly better, it is therefore used in our experiment.

C. Local update rule

When finding a set of data replicas, the ants use a local pheromone updating rule, which they apply immediately after having accessed replica j , as shown in (14).

$$\tau_j = (1 - \xi)\tau_j + \xi\tau_0, \forall j \in DT^i. \quad (14)$$

where $\xi (0 < \xi < 1)$ and τ_0 are two parameters. At the beginning of the selection process, a constant amount of pheromone is assigned to all the replicas, namely, $\tau_j = \tau_0 = CP$ (CP is a constant, $\forall j \in DT^i$). The local updating rule will reduce the pheromone trail of replica j after an ant visits it. In other words, it allows an increase in the exploration of replicas that have not been visited yet and, in practice, has the effect that the algorithm would not show stagnation behavior [8].

The data replica selection algorithm based on ACS is given in Algorithm 1 on the next page. The time complexity of the algorithm is $O(noa * k * l)$, where noa , k , and l denote the number of ants, the number of data sets, and the number of data replicas for each data sets, respectively.

Algorithm 1 Data replica selection based on ant colony system algorithm

Input:*MaxIt*: the maximum number of iterations;*noa*: the number of artificial ants; DT^i : the required data sets;**Output:***D*: a set of data servers to access each data set;

```
1:  $D = \emptyset; step = 0; \tau_0 = CP;$ 
2: while  $step < MaxIt$  do
3:    $step = step + 1;$ 
4:   set all ants at service endpoint;
5:    $D_{ant} = \emptyset;$ 
6:   for each ant  $k$  do
7:      $as = \emptyset;$  // data server list for each ant
8:     while ant  $k$  does not come back do
9:       ant  $k$  chooses a replica according to the state
       transition rule (11);
10:      update data server list  $as$ ;
11:      apply the local updating rule (14);
12:    end while
13:    if  $U(as) > U(D_{ant})$  then
14:       $D_{ant} = as;$ 
15:    end if
16:  end for
17:  when all ants return to service endpoint again, apply
  global updating rule (13) to  $D_{ant}$ ;
18:  if  $U(D_{ant}) > U(D)$  then
19:     $D = D_{ant};$  // keep the global-best set of data servers
    to  $D$ ;
20:  end if
21: end while
22: return  $D.$ 
```

VI. EXPERIMENTS AND ANALYSIS

The values of parameters considered in this paper are: $\alpha = 1$, $\beta = 2$, $q_0 = 0.9$, $\tau_0 = 0.1$, $CP = 0.1$, $\rho = 0.1$, $\xi = 0.1$, $\lambda_1 = 0.5$, $\lambda_2 = 0.5$, $noa = 10$, $MaxIt = 1000$. The performance of the proposed algorithm is affiliated to the size of the data replica selection problem. The size of the problem depends on the number of data sets required by each service and the number of data replicas for each data set. Thus, we generate two test groups. The first test group includes 4 test scenarios with different numbers of data replicas. The number of data replicas for each data set ranges from 3 to 6, in increments of 1. The number of required data sets is fixed at 10. The second test group includes 4 test scenarios with different numbers of required data sets. The number of data sets is 15, 20, 25 and 30. The number of data replicas for each data set is fixed at 6. This two test groups are designed to test how the running time of the proposed algorithm will change as the number of data sets and the number of data replicas change.

All test scenarios are run twenty times and the average values are reported. The price of a data replica, the network bandwidth (Mbps) between each data server and service endpoint, and the storage media speed (Mbps) are randomly drawn from a uniform distribution of the interval [1,100]. The

size (MB) of a data set is randomly drawn from a uniform distribution of the interval [1000,10000]. The number of data request in the waiting queue is randomly drawn from a uniform distribution of the interval [1,10]. These values will not affect the running time of the proposed algorithm, so their values can be in any intervals.

The simulation results are shown in Fig. 3 on the next page. Fig. 3(a) to Fig. 3(d) show the results of the first test group. Fig. 3(e) to Fig. 3(h) show the results of the second test group. In Fig. 3(a) to Fig. 3(h), the blue line denotes the utility of the best selection from the beginning of the trial, and the red point denotes the utility of the best selection of each iteration. The value of ‘GUtility’ is the utility of the best selection from the beginning of the trial and it depends on the cost and response time of data sets. That is to say, the change of the value of ‘GUtility’ has no significance for the simulation results. The value of ‘FRIT’ is the number of iterations when the best utility appeared and from this iteration the value of the best utility will not change.

According to the value of ‘FRIT’ in Fig. 3(a) to Fig. 3(d), as the number of data replicas increases from 3 to 6 in increments of 1, the ants need more iteration times to find the best selection. Fig. 3(i) shows the effect of the number of data replicas on the running time for finding the best selection. According to the value of ‘FRIT’ in Fig. 3(e) to Fig. 3(h), as the number of required data sets increases, the number of iterations also increases. When the number of required data sets is 15, 20, 25, and 30, the number of iterations to find the best selection is 27, 30, 40 and 56. All the figures in Fig. 3 show the proposed algorithm can solve the data replica selection problem efficiently.

VII. CONCLUSIONS

In our AMS data processing center, service composition involves large amounts of data being transferred, organized and stored, which makes the economics of the whole composition more complex. It is critical and challenging to be cost and time aware to manage the data and services in this intensive research environment. As data play the dominant role in execution of data-intensive service composition, the cost and response time of a data set influence the quality of the service that requires the data set. In the process of data-intensive service composition, service providers need to access a variety of data sets and pay for what they use. Service providers should make decisions about how to pay for the data sets and where to access them based on the number of service requests. They will always try to trade-off costs and gains when providing services. Thus, service providers need a data replica selection strategy, which can select the best replicas in order to reduce the response time and cost of their services. A novel data replica selection algorithm based on ant colony optimization is proposed, and the performance of the algorithm has been studied by simulations. The experimental results show that the proposed algorithm can solve data replica selection problem efficiently. Meanwhile, if the service endpoint has space to store a local copy of the data set and this is economically beneficial, a caching function will be invoked. To design a caching function for each data replica agent is also a part of our future work.

(a) Each data set has 3 data replicas

(b) Each data set has 4 data replicas

(c) Each data set has 5 data replicas

(d) Each data set has 6 data replicas

(e) Each service needs 15 data sets

(f) Each service needs 20 data sets

(g) Each service needs 25 data sets

(h) Each service needs 30 data sets

(i) The effect of the number of data replicas on the running time for finding the best set of data replicas.

Fig. 3: The results of the two test groups

REFERENCES

- [1] D. J. Abadi, "Data management in the cloud: limitations and opportunities", *IEEE Data Engineering Bulletin*, Vol. 32, No. 1, pp. 3-12, 2009.
- [2] H. H. E. Al-Mistarihi, and C. Yong, "Response time optimization for replica selection service in data grids", *Journal of Computer Science*, Vol. 4, No. 6, pp. 487-493, 2008.
- [3] R. M. Almuttairi, R. Wankar, A. Negi, R. R. Chillarige, and M. S. Almahna, "New replica selection technique for binding replica sites in Data Grids", *First International Conference on Energy, Power and Control*, pp. 187-194, 2010.
- [4] W. H. Bell, D. G. Cameron, R. Carvajal-Schiaffino, A. P. Millar, K. Stockinger, and F. Zini, "Evaluation of an economy-based file replication strategy for a data grid", *Proceedings of 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid*, pp. 661-668, 2003.
- [5] F. Casati, M. Sayal, and M. C. Shan, "Developing e-services for composing e-services", *Proceedings of 13th International Conference on Advanced Information Systems Engineering(CSiSE)*, Interlaken, Switzerland, Springer Verlag, pp. 171-186, 2001.
- [6] R. S. Chang, and P. H. Chen, "Complete and fragmented replica selection and retrieval in data grids", *Future Generation Computer Systems*, Vol. 23, No. 4, pp. 536-546, 2007.
- [7] F. Dong, J. Luo, A. Song, J. Cao, and J. Shen, "An effective data aggregation based adaptive long term CPU load predictions mechanism on computational grid", *Future Generation Computer Systems: the international journal of grid computing: theory, methods and applications*, Vol. 28, No. 7, pp. 1030-1044, 2013.
- [8] M. Dorigo, and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem", *IEEE Transactions on Evolutionary Computation*, Vol. 1, No. 1, pp. 53-66, 1997.
- [9] Z. Fei, S. Bhattacharjee, E. W. Zegura, and M. H. Ammar, "A novel server selection technique for improving the response time of a replicated service", *Proceedings of Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies*, Vol. 2, pp. 783-791, 1998.
- [10] C. Ferdean, and M. Makpangou, "A scalable replica selection strategy based on flexible contracts", *Proceedings of the The Third IEEE Workshop on Internet Applications*, pp. 95-99, 2003.
- [11] J. D. Guyton, and M. F. Schwartz, "Locating nearby copies of replicated Internet servers", *ACM SIGCOMM Computer Communication Review*, Vol. 25, No. 4, pp. 288-298, ACM, NY, USA, 1995.
- [12] J. S. Gwertzman, and M. Seltzer, "The case for geographical push-caching", *Proceeding of the 5th Workshop on Hot Topics in Operating Systems*, pp. 51, IEEE Computer Society, Washington, DC, USA, 1995.
- [13] T. Ho, and D. Abramson, "The griddles data replication service", *Proceeding of the 1st International Conference on E-Science and Grid Computing (E-Science'05)*, pp. 271-278, 2005.
- [14] W. Hoschek, J. Jean-Martinez, A. Samar, H. Stockinger, and K. Stockinger, "Data management in an international data grid project", *Proceedings of the First IEEE/ACM International Workshop on Grid Computing*, pp. 77-90, Springer-Verlag, UK, 2000.
- [15] A. Ismail, J. Yan, and J. Shen, "Incremental service level agreements violation handling with time impact analysis", *Journal of Systems and Software*, Available online 18 Feb, ISSN 0164-1212, 2013.
- [16] S. Kulkarni, N. Ganguly, G. Canright, and A. Deutsch, "A new bio-inspired location search algorithm for peer to peer network based Internet telephony", *Proceeding of the 1st International Conference on Bio Inspired Models of Network, Information and Computing Systems*, pp. 1-5, 2006.
- [17] M. Milenkovic, E. Castro-Leon, and J. R. Blakley, "Power-aware management in cloud data centers", *CloudCom*, LNCS 5931, pp. 668-673, 2009.
- [18] V. M. Munoz, G. A. Vicente, F. G. Carballeira, and J. S. Cairols, "Emergent algorithms for replica location and selection in data grid", *Future Generation Computer Systems*, Vol. 26, No. 7, pp. 934-946, 2010.
- [19] J. Niemoller, R. Quinet, R. Levenshteyn, and I. Fikouras, "Cost control in service composition environments", *Proceedings of the Second International Conference on Next Generation Mobile Applications, Services and Technologies (NGMAST '08)*, pp. 25-32, 2008.
- [20] R. M. Rahman, K. Barker, and R. Alhajj, "A predictive technique for replica selection in grid environment", *Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2007)*, pp. 163-170, 2007.
- [21] R. M. Rahman, K. Barker, and R. Alhajj, "Replica selection strategies in data grid", *Journal of Parallel and Distributed Computing*, Vol. 68, pp. 1561-1574, 2008.
- [22] K. Ranganathan, and I. Foster, "Identifying dynamic replication strategies for a high-performance data grid", *Proceedings of the Second International Workshop on Grid Computing*, pp. 75-86, 2001.
- [23] M. Ripeanu, and I. Foster, "A decentralized, adaptive replica location mechanism", *Proceedings of the 11th IEEE International Symposium on High Performance Distributed Computing*, pp. 24-32, IEEE Computer Society, Washington, DC, USA, 2002.
- [24] M. Sayal, Y. Breitbart, P. Scheuermann, and R. Vingralek, "Selection algorithms for replicated web servers", *SIGMETRICS Perform. Eval. Rev.*, Vol. 26, No. 3, pp. 44-50, ACM, NY, USA, 1998.
- [25] M. Sayal, P. Scheuermann, and R. Vingralek, "Content replication in Web++", *Proceedings of the Second IEEE International Symposium on Network Computing and Applications*, pp. 33, IEEE Computer Society, Washington, DC, USA, 2003.
- [26] J. Shen, G. Beydoun, S. Yuan, and G. Low, "Comparison of bio-inspired algorithms for peer selection in services composition", *IEEE International Conference on Services Computing (SCC 2011)*, pp. 250-257, 2011.
- [27] Y. Song, H. Wang, Y. Li, B. Feng, and Y. Sun, "Multi-Tiered on demand resources scheduling for VM-based data center", *IEEE/ACM International Symposium on Cluster Computing and the Grid*, pp. 148-155, 2009.
- [28] M. Sun, J. Sun, E. Lu, and C. Yu, "Ant algorithm for file replication selection in data grid", *Proceedings of the First International Conference on Semantics, Knowledge and Grid*, pp. 64-66, 2005.
- [29] S. Vazhkudai, S. Tuecke, and I. Foster, "Replica selection in the globus data grid", *Proceedings of the 1st International Symposium on Cluster Computing and the Grid*, pp. 106-113, IEEE Computer Society, Washington, DC, USA, 2001.
- [30] S. Venugopal, and R. Buyya, "An SCP-based heuristic approach for scheduling distributed data-intensive applications on global grids", *Journal of Parallel and Distributed Computing*, Vol. 68, No. 4, pp. 471-487, 2008.
- [31] L. Wang, C. Di, Y. Li, J. Shen, and Q. Zhou, "Towards minimizing cost for composite data intensive services", *IEEE 17th International Conference on Computer Supported Cooperative Work in Design*, accepted in February, 2013.
- [32] L. Wang, and J. Shen, "Towards bio-inspired cost minimization for data-intensive service provision", *IEEE First International Conference on Services Economics*, pp. 16-23, 2012.
- [33] L. Wang, J. Shen, and J. Yong, "A survey on bio-inspired algorithms for web service composition", *Proceeding of the 2012 IEEE 16th International Conference on Computer Supported Cooperative Work in Design*, pp. 569-574, 2012.
- [34] M. Winter, "Data center consolidation: A step towards infrastructure clouds", *CloudCom*, LNCS 5931, pp. 190-199, Springer, Heidelberg, 2009.
- [35] K. P. Yoon, and H. C. Land, "Multiple attribute decision making: an introduction", *Sage university papers series (quantitative applications in the social sciences)*, Thousand Oaks, CA, Sage Publications, 1995.
- [36] E. W. Zegura, M. H. Ammar, Z. Fei, and S. Bhattacharjee, "Application-layer anycasting: a server selection architecture and use in a replicated web service", *IEEE/ACM Transactions on Networking*, Vol. 8, No. 4, pp. 455-466, 2000.
- [37] Y. Zhao, and Y. Hu, "GRESS: A grid replica selection service", *Proceedings of 16th International Conference on Parallel and Distributed Computing Systems*, 2003.