Future Generation Computer Systems 29 (2013) 953–972

Contents lists available at SciVerse ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

A two phased service oriented Broker for replica selection in data grids

Rafah M. Almuttairi [a,*], Rajeev Wankar [a], Atul Negi [a], C.R. Rao [a], Arun Agarwal [a], Rajkumar Buyya [b]

[a] Grid Computing and Distributed Systems Lab, Department of Computer and Information Sciences, University of Hyderabad, Hyderabad, India

[b] Cloud Computing and Distributed Systems Lab, The University of Melbourne, Parkville, Australia

ARTICLE INFO

Article history:
Received 26 October 2011
Received in revised form
16 September 2012
Accepted 26 September 2012
Available online 30 October 2012

Keywords:
Data grids
Replica selection Broker
Selection technique
Resource Broker
Association rules

ABSTRACT

Replica selection is one of the fundamental problems in *Data Grid*'s environment. This work's concern is designing a *Two phased Service Oriented Broker* (2*SOB*) for *replica selection*. It is focused on investigating, selecting, modifying, and experimenting with some *non-conventional approaches* to be applied on the relevant *selection techniques*. The motivation of this work is to introduce a *novel* Service Oriented*Broker for Replica Selection in Data Grid*. The main characteristics of this *2SOB* are: *Scalability*, *Reliability, Availability, Efficiency* and *Ease of deployment*. *2SOB* consists of two phases; the first is a *Coarse-grain phase,* basically used for sifting replica sites that have low latency (uncongested network links) and distinguishing them from other replicas having a high latency (congested network links). Procedurally, this has been done using the *association rules* concept of the *Data Mining* approach. The second is a *Fine-grain phase,*used for extracting the replicas admissible for user requirements through applying *Modified Minimum Cost and Delay Policy (MMCD)*. Both phases have accordingly been designed, simulated, coded, and then validated using real data from *EU Data Grid.*The first phase has thereby been applied on the real network data of *CERN* (*February* 2011). Experimentations compared with some other contemporary selection methods of different Brokers showed appreciable results. Using this proposed Broker it is possible to achieve an enhancement in the speed of executing Data Grid jobs through reducing the transfer time.

1. Introduction

The *Large Hadron Collider (LHC)* is a good example of scientific collaboration where massive number of computing elements is used to process, store and share extremely large data collections. This is achieved by building Internet-based distributed computing platforms such as, *LHC Computing Grid (LCG)*. In *LHC*, the data is replicated into additional sites around the world to improve data access performance and to reduce the latency of data transfers as shown in Fig. 1. *CERN* provides *Data Grid* applications. These applications allow people who work in the particle physics field to invent and run simulations as well as generate, test, and re-test experiments many times.

This is possible through the construction of LCG. LHC researchers use the Internet to download a large amount of data, perhaps up to 1 TB from distant sites. The condition of network links show variations in terms of latency and other QoS parameters depending upon the time of the day or week, and also the replica site location. Using Data Grid middleware services such as Replica Location Service (RLS), Network Monitoring service (NMS) and Data Transport Service (GridFTP) [1], a service oriented Broker can discover all available replicas and select the best set of replicas that matches the user's requirements [2,3].

When the user/application has a constraint of squeezing a job into an execution time slot the selection decision becomes a critical decision, because it affects the performance and total job execution time. This is generally called a *Replica Selection Decision* in Data Grids [4,5].

A *Data Grid* in the main is a service oriented platform where all kind of resources (files) are treated as services to users. The main aim of Data Grid architecture is providing a scalable infrastructure for the management of storage of the resources and data which are distributed across the grid environment. Data Grid also provides a reliable service with an easy and scalable access to all those who search for services on the Internet. These services are designed to support data intensive applications that require access to the large amount of data (terabytes or even petabytes) with the varied quality of service requirements. By utilizing a set of core services, such as data transport, replica cataloguing, and network monitoring service, various high-level services, such as brokering, can be constructed.

In this work, the design and implementation of a high level brokering service that can be used with Data Grids is discussed. The proposed Broker uses information regarding replica location and user preference to guide selection from among storage

* Corresponding author. Tel.: +964 91988525407, +964 7810852588.
E-mail address: rafahmohammed@gmail.com (R.M. Almuttairi).

0167-739X/$ – see front matter
doi:10.1016/j.future.2012.09.007

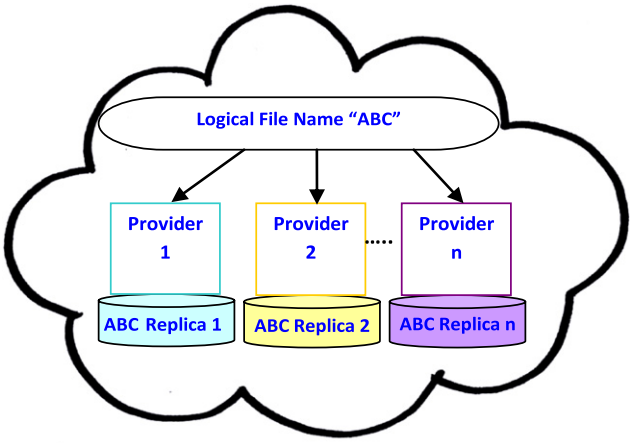


Fig. 1. Replicas of ABC file.

replica alternatives in order to ensure reliable bulk data transfer with constrained time requirements. Extracting multiple replica providers that are stable and having a closest match to the user's requirements is one solution to this problem. To reach this objective, a two-phased service oriented Broker has been proposed with two phases of selection policy to reduce transfer time by discovering, selecting, reserving and assigning the best associated providers. 2*SOB* uses an association rules concept of the Data Mining approach as an associated replicas discovery process.

Data Mining refers to non-trivial extraction of implicit, previously unknown, and potentially useful information from data. This encompasses a number of technical approaches, such as clustering data summarization, classification, finding dependency networks, analyzing changes, and detecting anomalies. Most of these approaches have been used in replica selection techniques proposed in the literature. All the previous methods of the above mentioned approaches are not so convenient for *Broker design* to achieve *replica selection* in Data Grids environments such as *LCG*, for example. This is due to large computational complexity and high I/O operations requirement. Those methods generally differ from each other in their architecture, computational cost, and quality [6].

The *Replica Selection Decision* in earlier work such as Lin et al. [5], shows a single replica site that is selected based on a variable pricing scheme. To handle growing amounts of the large file transfers in a graceful manner, the ability of a Broker and the network monitoring service is enhanced [7]. In this work, scalability is covered by expanding the Broker to support selection of a set of replica sites using Data Mining approach. These sites work collaboratively to transfer data files/sets in parallel [8].

Here we propose a novel approach using association *rules*. *Association rules* concept is used to find group of associated service providers (replica sites) which have lower rates of packet drops with good latency at the time of data file transfer [9]. Our new approach has two phases which are:

Phase one: we propose Coarse-grain selection criteria: Sifting replica sites that have low latency (uncongested links).
Phase two: we propose Fine-grain selection criteria: Extract the associated replica sites that have lowest prices or any other *QoS* of user requirements.

The associated sites can collaborate to share the transfer of the large file by dividing the file into multiple parts among themselves so that each replica site can send a part of it.

Our experimental data has been taken from the *National Accelerator Laboratory*, *SALC* where sites of the world are being seen from *CH.CERN.N20* [10,11].

The rest of the paper is organized as follows: In Section 2, related work is reviewed. In Section 3, a preliminary concept and parallel between Data Grids and Data Mining is presented. In Section 4, two phased service oriented architecture is presented and the formulation of the path determination problem is described. Section 5 presents the proposed selection policies of two phases. The performance evaluation and discussion of the results are presented in Section 6. Comparisons with other methods while selecting single and set of providers are made in Sections 7 and 8 respectively. Performance and results are given in Section 9. Constraints and limitations of the proposed work are explained in Section 10. Finally, the Discussion and conclusions are given in Section 11.

2. Background and related work

A Broker is a decision making tool that determines how and when to acquire grid services and resources for higher level components. To get the best *QoS* of user's requirement, when the job execution time is one of its constraints, the Broker has to optimize its work using a smart replica selection strategy. Replica selection strategy is an important part of the decision-making in the Broker of Data Grid [12].

In the last few years there were two heuristic directions followed by researchers to reduce the total time of the execution of the Data Grid jobs. The first approach is to *reduce the selection time* (the consumed time when the Broker receives the request till the best provider is selected). The second approach is *to reduce the file transfer time* (the consumed time since the provider has been decided till the file is transmitted).

Using the first approach researchers proposed different selection strategies that can be used in the Broker to enhance the selection process [5]. Using the second approach the problem has been investigated by developing a co-allocation architecture and loading balancing algorithm [13] to deal with fluctuating transfer rate, in order to enable parallel downloading of datasets from multiple servers. The objective of the second approach is to exploit rate differences among various client–server links and to address dynamic rate fluctuations by dividing files into multiple blocks of equal sizes to improve the performance of data transfer in Data Grids [14]. A service oriented Broker adapts its selection criteria dynamically so that the best approximate application providers are matched with clients' requirements [13].

In this work, a novel replica selection Broker with new strategies is proposed to cover the objectives of both approaches (reduce the time of selecting the provider and reduce the file transfer time) [13]. The first phase is used to generate sets of replicas (providers) [8,13]. A best set of replica provider sites is obtained by the second phase. The best providers concurrently send different parts of the files [15–17] to the computing sites.

This section presents some selection strategies which are proposed in the literature to improve the performance of a resource (replica) selection Broker.

Gwertzman and Seltzer [18] and Guyton and Schwartz [19] proposed replica selection approaches based on binding a client to the nearest replica, with respect to some static metric such as the geographical distance in miles and the topological distance in number of hops [20]. However, as several experimental results [21,22] show, the static metrics are not good predictors for the expected response time of client requests. The main drawback of both geographical and topological network metrics is that they ignore the network path's dynamic conditions. From a different focus, Kavitha and Foster [23], used a traditional replica catalog based model; for each new request, Replica Location Service is queried to get the addresses of replica's sites and then the network link is probed using the Hop count method to select the best

replica. The drawback of this approach is that it depends on the number of hops that may not reflect the actual network condition such as Network Bandwidth and link's latency. On the other hand, Vazhkudai et al. [24,25] contributed in many research results. In their work they used the history of previous file transfer information to predict the best site holding a copy of the requested file. When a file transfer has been made between two sites, the file size, the available network bandwidth, and transfer time are saved. Thus it can be used later for training and testing the regression model to predict the actual transfer time. In their work they showed that data from various sources can help in better predictions than data from one source. They achieved a better accuracy in file transfer throughput prediction by using data from all of these three sources: network data streams, file size, and past grid transfer information.

Rahman et al. [26] exploited a replica selection technique with the *K-Nearest Neighbor (KNN)* rule used to select the best replica from the information gathered locally. The *KNN* rule selects the best replica for a file by considering previous file transfer logs indicating the history of the file and those similar. This technique has a drawback as they mentioned in their paper: the misclassification will increase in case of the large file transfer and will cost more than a couple of small file transfers, especially in the Gaussian random access pattern where the accuracy is the lowest. Another drawback in *KNN* is that one needs to save all previous instances (file requests) to use them to select the best replica site, which means it will take some time to search in the large history of the database and the result might or might not be correct. Rahman et al. [3] also proposed a *Neural Network* predictive technique (NN based) to estimate the transfer time between sites. The predicted transfer time can be used as an estimate to select the best replica site among different sites. On the other hand, Jadaan et al. [15] proposed a new selection technique that uses a *Rank based Genetic Algorithm* [15]. The main objective of using GA is to reduce the time of searching for an appropriate replica provider who has a close matching to the user/application request. GA is used as a clustering method.

Lin et al. [5] have explored the effectiveness of economy-based resource management in Data Grids and proposed a policy for a data replication Broker that improved replication. An economic data resource management system was shown to perform with marked improvements when using a variable pricing scheme that is based on the expected actions of a rational agent compared to a fixed pricing scheme. The policy displayed an effective means of improving the performance of the grid network traffic as was indicated by the improvement of speed and cost of transfers by Brokers.

Litzkow et al. [27] proposed the high-throughput computing platform called Condor, wherein a central manager is responsible for matching resources against jobs. Obvious disadvantages to this approach are scalability and single point of failure.

All previous strategies were looking to find the best single replica site using different approaches. Factors of transfer quality such as node-to-node link reliability, actual transfer times, and server reliability have been ignored so that future Broker decisions will make the same choices for similar jobs even though the network conditions have changed. In general the previous methods lack scalability for the transfer of large files in constrained time. To overcome this scalability problem, a method to find the dependency of replicas has thus been developed. A developed selection strategy is used to select a set of replica sites and utilize these sites to fetch the huge data files in a collaborative or parallel manner. In our previous works [8,28,29,16,17,13,30] the *association rules* of the Data Mining approach were used to enhance the replica selection strategy. In this work a complete technique of the novel replica selection Broker with a real data implementation (from the *CERN* project) has been explained.

Table 1
Example database with 4 items and 5 transactions.

Trans. ID	Milk	Bread	Butter	Beer
1	1	1	0	0
2	0	0	1	0
3	0	0	0	1
4	0	1	0	0
5	0	1	0	0

3. Preliminary concepts

In this section the concepts of Data Mining and Data Grid with the common convergence between them are explained.

3.1. Notation in Data Mining (DM)

This section presents a methodology known as association rule mining, useful for discovering interesting relationships hidden in huge data sets. Association rules have received lots of attention in DM due to their many applications in marketing, advertising, inventory control, and many other areas. Association Rules can be derived using supervised and unsupervised processes [31]. Following the definitions by Agrawal et al. [32], the problem of association rule mining is explained in the following:

Database (D): Let $I = \{i_1, i_2, \ldots, i_n\}$ be a set of n binary attributes called items. Let $D = \{t_1, t_2, \ldots, t_m\}$ be a set of transactions called the database. Each transaction in D has a unique transaction *ID* and contains a subset of the items in I.

An Association Rule (AR): AR is a rule that is defined as an implication of the form $X \rightarrow Y$, where $X, Y \subseteq I$, and, $X \cap Y = \emptyset$, The sets of items (for short itemsets) X and Y are called antecedent (left-hand-side or LHS) and consequent (right-hand-side or RHS) to the rule respectively. To illustrate the concepts, we use a small example from the supermarket domain. The set of items is I = milk, bread, butter, beer and a small database containing the items (1 codes presence and 0 absence of an item in a transaction) is shown in the Table 1. An example rule for the supermarket could be $\{butter, bread\} \rightarrow \{milk\}$ meaning that if *butter* and *bread* are bought, customers also buy *milk*.

Note: This example is an extremely small and trivial example. In practical applications, a rule needs the support of several hundred transactions before it can be considered statistically significant, and datasets often contain thousands or millions of transactions.

To select interesting rules from the set of all possible rules, constraints on various measures of significance and interest can be used. The best known constraints are minimum thresholds on support and confidence.

The support (s): The support of the itemset X is referred by $s(X)$. It is defined as the proportion of transactions in the data set which contain the itemset. In the example database, the itemset milk, bread, butter has the support of $(1/5 = 0.2)$ since it occurs in (20%) of all transactions (1 out of 5 transactions).

The confidence (c): The confidence of a rule is defined as: $c(X \rightarrow Y) = s(XUY)/s(X)$. For example, the rule $\{milk, bread\} \rightarrow \{butter\}$ has a confidence of $(0.2/0.4 = 0.5)$ in the database, which means that for (50%) of the transactions containing *milk* and *bread* the rule is correct. Confidence can be interpreted as an estimate of the probability $P(Y/X)$, the probability of finding the RHS of the rule in transactions under the condition that these transactions also contain the LHS [32,31].

The Improvement (lift): It is the lift of a rule that is defined as $I(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(Y) \times \sigma(X)}$ or the ratio of the observed support to that expected if X and Y were independent. The rule $\{milk, bread\} \rightarrow \{butter\}$ has a lift of $0.2/(0.4 \times 0.4) = 1.25$.

A Frequent Item Set (FIS): An itemset X is said to be a Frequent Item Set (FIS) in T with respect to (*min-supp*), iff $s(X) \geq$ *min-supp* [32].

A Maximal Frequent Set (MFS): A *FIS* is called Maximal Frequent Set (MFS) if no supper itemset of this itemset is a *FIS* [32].

3.2. Notation from Data Grids

Here briefly we review some notation from Data Grids that is being used to explain the concepts.

Replica provider Sites or Replicas (RS): It is a grid site(s) that has a storage element which is used to store copies of some files. In short it is called *Replicas. RS* or S_i, where, $i = \{1, 2, \ldots, M\}$, and M is the maximum number of replicas. Each replica provider site S_i has a vector of attributes determined by the administrator of the site which is denoted by: $S_{i,j}$, where, $j = \{1, 2, \ldots, N\}$, and N represents the maximum number of attributes for each replica.

Replica: A copy of requested file (f).

User Request (UR): Ii is vector of attributes denoted by: a_h, where, $h = i = \{1, 2, \ldots, Q\}$, and Q is the subset of user attributes and $(Q < N)$.

Network History File (NHF): It is a file that contains a complete history of values of Round Trip Time (RTT) or Single Trip Time (STT) of connected replicas at interval time $[t_0 - t_z]$. This file can be formed using a monitoring tool such as *PingER* [10], for example NHF contains:

- *Rows* = *Transactions (T) within an interval time* $[t_0 - t_z]$.
- *Columns* = *Identification of Replica Sites (IDRS)*, S_i.

Session: Let *NHF be* divided into f sessions (partitions) at different slots of time, $P = \{P_1, P_2, \ldots, P_f\}$, where P_f is called the latest session of NHF. It is the most recent transaction within the time interval $[t_{z-l} - t_z]$, and l is the size of interval of *Pf in NHF* which is denoted by *NHF* $[P_f]$.

Data Grid Job (J): A job contains a set of data files which need to be accessed and analyzed.

The above mentioned concepts of Data Mining are applied on the information from the Data Grid to mine the data about the links which connect replicas. The concept of Data Grid Mining is new and for the first time is used in this paper. This novel idea is discussed in more detail in the next section where the parallels between Data Mining concepts and Data Grid concepts are drawn out.

3.3. Parallels between Data Grid and Data Mining

Before explaining the present approach it might be convenient or rather essential to review some concepts about the parallelism between Data Grid and Data Mining techniques.

Here we synthesize the novel concept of Data Grid Mining, a type of Data Mining which may be generally useful in assessment and management of Data Grid systems.

In this work we use it in a very specific instance, that is to find out relationships between the instantaneous network conditions between replicas. It takes advantage of the huge amount of information gathered by a network monitoring service to look for patterns in replicas' link behavior. The parallelism between Data Mining and Data Grid Mining concepts is sketched out in the following aspects:

Item: An item in Data Mining is an attribute value which represents the presence or absence of an attribute in a specific transaction; while in Data Grid Mining it is a Boolean value that determines whether or not the link is stable in a specific transaction.

Database (D): The *Network History File (NHF)* in *Data Grid Mining* represents the *Database* of *Data Mining*.

Transactions (T): A transaction in Data Mining is a set of items (attributes), $T = \{i_1, i_2, \ldots, i_n\}$, where n is number of attributes. In the case of Data Grid, each transaction (T_k) is a set of times in milliseconds of either single or round trips from replica providers to the computing site $T_k = \{RTT_1, RTT_2, \ldots, RTT_N\}$, where k is a slot time $0 \leq k \leq z$, and N is the number of replicas (providers).

Association Rules (AR): The purpose of mining association rules in a database is to discover hidden information between attributes. In Data Grid Mining the Transferring Rules (TR), the rules that represent the most likely relationships among links connecting replicas and the power of the cooperating among the replica provider sites, are represented by a CTR set.

Frequent Item Set (FIS): In Data Mining it represents a set of associated objects [6]. In Data Grid Mining it represents a set of replica provider sites located in the different locations with a similar behavior of their links (assumed to be uncongested) and known as *Associated Replica Sites* (ARS).

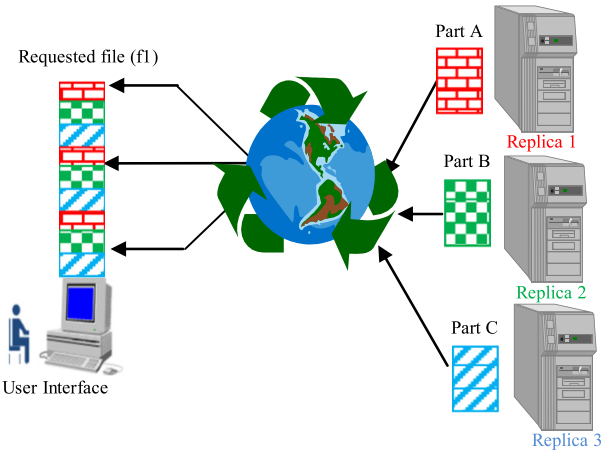


Fig. 2. Concurrent bulk data transfer of file (f) from multiple replicas.

4. Two-phased Service Oriented Broker Architecture (2SOB)

2SOB is used in Data Grid architecture to provide a scalable Data Grid management system. Traditionally, a centralized approach to resource management is used in the resource Brokers, wherein a single node is responsible for decision making. An example of such an environment is Condor [27]. Obvious disadvantages to this approach are scalability and a single point of failure. In Condor an efficient recovery mechanism is used to address failure and has been proven to scale to thousands of resources and users. Despite this, there is a more fundamental problem with this centralized approach when applied to Grids. In these highly distributed environments, there are numerous user communities and shared resources, each with distinct security requirements. No single resource Broker is likely to be trusted by all of these communities and resources with the necessary information to make decisions. At the extreme, each user may need his or her own Broker, because only that user has the authorization to gather all the information necessary to make brokering decisions. For this reason, we have designed a decentralized selection brokering strategy wherein every client that requires access to a replica performs the selection process rather than a central manager performing matches against clients and replicas as shown in Fig. 4.

In 2*SOB* we adopt a Grid structure based on a simplification of the architecture proposed by the EU Data Grid project [33]. The Grid consists of several sites, each of which may provide resources for submitted jobs. Computational and data-storage resources are called *Computing Elements (CEs)* and *Storage Elements (SEs)* respectively. The grid site that has Computing Elements is called *Computing Site (CS)*. CS runs jobs that use the data in files stored on Storage Elements. A Resource Broker controls the scheduling of jobs to Computing Elements.

Actually, the proposed Broker (2SOB) is useful when there is a need to retrieve different segments of data from multiple data sources as it is shown in Fig. 2.

2SOB is a *daemon* process that listens to a *TCP/IP* socket for client requests. Upon client interaction, a thread is spawned to handle the client messages using a new available port. The 2SOB

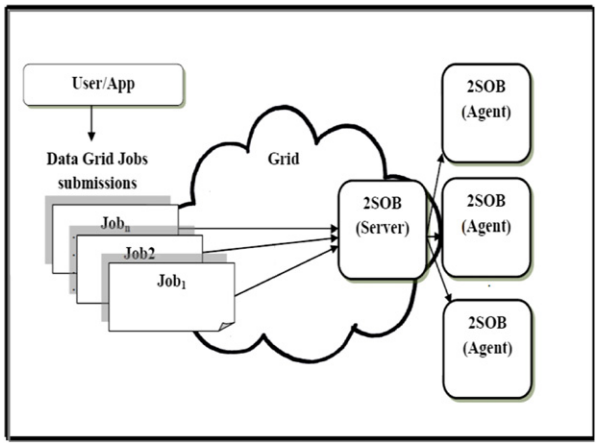


Fig. 3. Two phased Service Oriented Broker infrastructure.

master daemon acts as the preliminary Broker for *agent* thread to handle *individual* client. This section shows the main steps of the replica brokering algorithm which are used in 2SOB for the replica selection process. The actions of the daemon are described in the following:

Input:
One or more requests of Data Grid jobs.
Action:
Initiate an instance of 2SOB (agent) and assign it to each job as shown in Fig. 3.
Output:
Each job is assigned to a single agent.
The actions of the agent are described as:
Input:
A constraint execution time of the required Data Grid job.
Action:
Select the most appropriate replica provider(s) to transfer the required data.
Output:
Set of associated replicas that match user's requirements (see Figs. 5–7).

Here we provide in detail the behavior of each agent as shown in Fig. 4:

1. *Resource Requirement Interpreter*: It is used to interpret the requirements of the user/application.
2. *Replica Discovery Unit*: It is used to contact the *Dynamic Grid Resource Information Services (DGRIS)* to obtain a list of available replica providers with static and dynamic resource information (hardware and software characteristics, current queue and load, etc.). The service registry is a part of the service discovery component and it is similar to Universal Description, Discovery and Integration (UDDI). The registry is responsible for caching advertisement from available service with other maintenance information including the *semantic* information (e.g., the type of data the service handles and quality of service (QoS) that the service provides). The service description is stored as a set of attribute-value pairs [34].
3. *Replica Management*: It is used to receive network reports and a list of replica providers with their attribute values. This unit generates the NHF for the Coarse-grain phase and also generates a standardized table of replica providers attributes to be used by the Fine-grain phase.
4. *Associated Resource Discovery*: It is used to sift the set of replica providers having uncongested network links which are called Associated Replica Sites (ARS).
5. *Ranking and Filtering*: It is used to rank the associated replicas set to select Best Replica Sites (BRS).
6. *Resource Reservation Unit*: It is used to reserve the resources of BRS.
7. *Data Transport Service* (*DTS*): It is used to transfer the required file(s) from the selected replica servers to the computing site.

In this work, the main objective behind proposing a replica Broker is to select the "best" replica provider set. This practically means to set the number of sites with *good* latency (the amount of time taken by a packet to travel from source to destination), and to maintain good bandwidth to speed up and fully utilize the capacity (throughput) of a network [5].

A further objective of this work is to gain knowledge about the *operation kinetics* of the supporting underlying network. To reach this objective, 2*SOB* utilizes a network monitoring service; a core service to get a real picture of the network links latency. In a Data Grid infrastructure, the network monitoring service measures the latency using *Round Trip Time*.

In bulk data transfer the single trip from replica provider to computing site is required to be monitored. However, using roundtrip-based measurements, it is hard to isolate the direction in which congestion is experienced. Therefore in this work, a *One Way Ping Service OWPS* [35] is suggested to be added in the network monitoring service. *OWPS* is used to get *Single Trip Time (STT). STT* is a time taken by the small packet to travel from *replica provider* to the computing site. The *STT* delays include five types of delays which are: packet-transmission from *replica provider* to computing site delays, the transmission rate out of each router and out of the replica site, packet-propagation delays (the propagation on each link), packet-queuing delays in intermediate routers and switches, and packet-processing delays (the processing delay at each router and at the replica site) for a single trip starting from *replica provider* to *computing site*.

5. Selection process

Selection process is the main task of 2SOB. As stated earlier, the proposed selection strategy has two phases: Coarse-*grain* phase and *Fine-grain* phase which are elaborated on in the next subsections.

5.1. Coarse-grain phase

Coarse-grain phase uses the association rules concept upon data sets of the Network History File [8]. The association rules algorithm is the process of finding hidden relationships in data sets and summarizing these patterns in models. It is meant for combing through data to identify patterns and establish relationships that allow an applicant to discover knowledge that is hidden in large data sets [8]. The algorithm is looking for patterns where one event is connected to another event. In our model, the Coarse-grain phase is when the problem of discovering the association rules in Data Grids can be decomposed into two sub problems:

The first is finding Frequent Item Sets (FIS). The Maximum Frequent Set (MFS) in our context is known as Associated Replicas Sites (ARS). This can be done using Efficient Set Technique (EST) [13].

The second is deriving the Transferring Rules (TR) from the Frequent Item Sets (FIS) [13]. Table 2 lists the notations used to write the algorithms of 2SOB.

5.1.1. Efficient Set Technique (EST)

The first stage of the Coarse-grain phase is to find all Frequent Item Sets. The algorithms which are used to do that are explained as the following:

5.1.1.1. Mapping Function (MF). MF is used to convert the Network History File (NHF) into the Logical History File (LHF). The LHF is

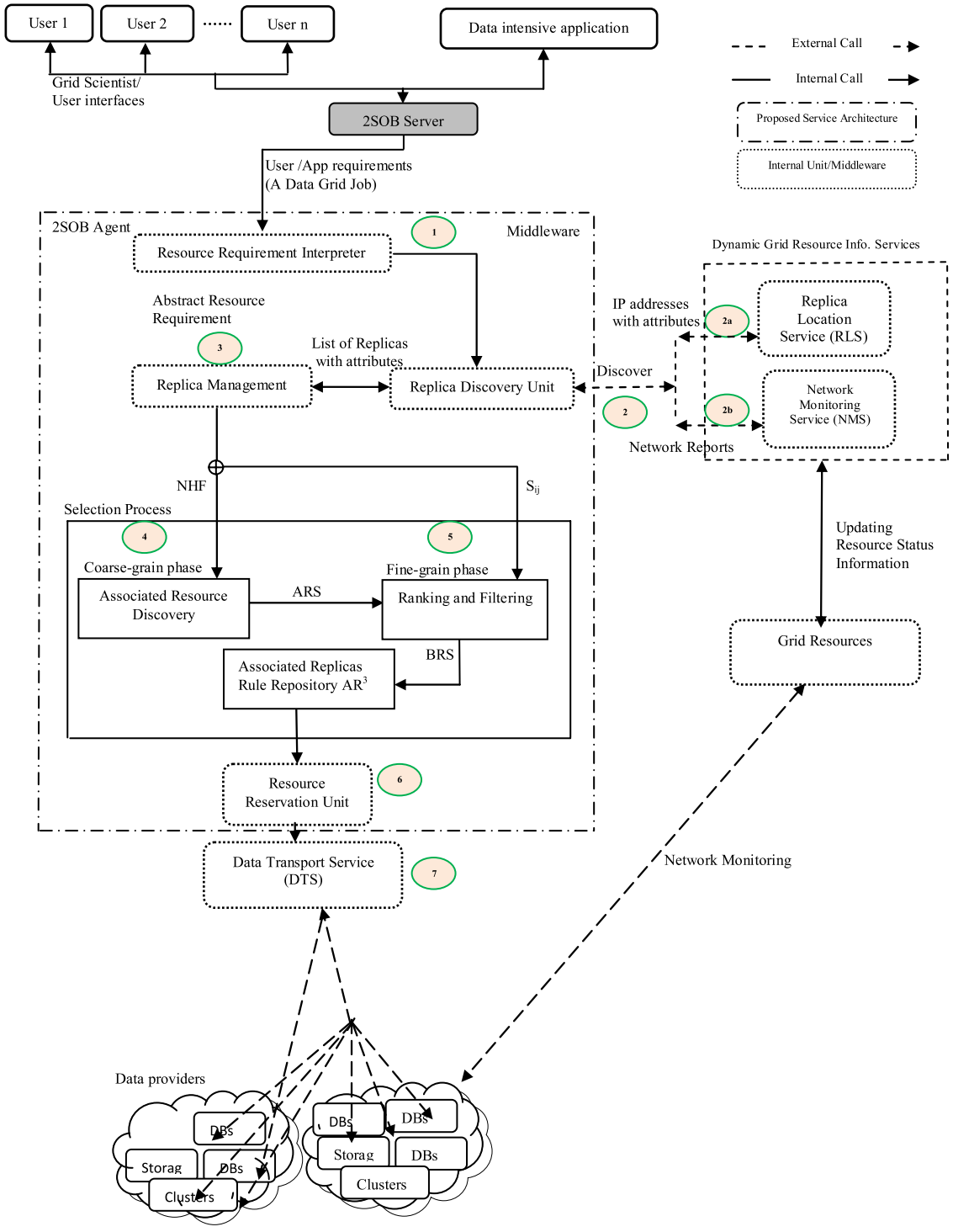


Fig. 4. Architecture of 2SOB Broker.

a table containing Boolean values (0, 1). The mapping algorithm shown in Fig. 8 is applied on the round trip time (RTT) or Single Trip Time (STT) between computing site and replicas in an interval time $[t_0 - t_z]$ as shown in Tables 3 and 4.

5.1.1.2. Generate frequent item sets technique (FIST). In this section, one of the association techniques of Data Mining is used to generate a set of associated replicas. The *Apriori* algorithm [6] is used here to discover the hidden relationships among network links of the replica providers, such as latency, for example. The result of applying the Apriori algorithm is a set of replica providers having similar characteristics of links conditions (set of replica providers having uncongested links) at the time of job execution. The associated replicas are also called the *Frequent Item Sets* as shown in Table 5b. The generated Frequent Item Sets are used to generate the association rules. The algorithm in Fig. 9 shows the main steps of the generation of Frequent Item Sets (FIS).

Fig. 10 explains the main idea of the algorithm, in *line* 1 *LHF* is analyzed to generate the candidate and frequent 1-*itemset*. This is done by calculating the support of each data item and comparing it to the minimum support. A loop from *line* 4 to *line* 20 in the algorithm is used to generate all Frequent Item Sets which are,

Table 2
List of notations.

	Notation	Description
1	STT	Time taken by single trip
2	RTT	Time taken by round trip
3	FIS	Frequent Item Set (replica sites)
4	MFS	Maximum Frequent Item Set
5	MR	Maximum number of items in a FIS
6	T	Transactions of an interval time $[t_0 - t_z]$
7	NHF	NHF: Network History File
8	ARS	List of Associated Replicas Sites ARS_j, where $j = \{1, 2, \ldots, m\}$, m represents number of associated sites and $m \leq M$, M represents total Replicas Sites.
9	BRS	Best Replica Set BRS_i, where $i = \{1, 2, \ldots, W\}$, W represents Number of Replicas in BRS, $W < m$
10	C_k	Set of candidate of k-itemsets
11	L_k,	Set of large k-itemsets with min-supp
12	l_j, l_j	A frequent $(k-1)$-itemsets within L_{k-1}
13	$l_i[m]$	It is a m-th item in itemset l_i
14	min-conf	Minimum confidence value
15	min-supp	Minimum support value
16	NR	Maximum number of rules in a TR
17	s	Support value
18	c	Confidence value
19	$S_{i,j}$	Replica provider site
20	M	Max. Number of replica providers
21	N	Max. Number of attributes in a Site
22	CTR	Correlated Transferring Rules
23	k-itemset	An itemset with k items
24	TR	Transferring Rules
25	AR3	Associated Replicas Rule Repository
26	LHF	Logical History File with (0, 1) values
27	EST	Efficient Set Technique
28	OWPS	One Way Ping Service
29	DGRIS	Dynamic Grid Resource Information Service
30	TRT	Transferring Rule Technique
31	CTRT	Correlation Transferring Rule Technique
32	FIST	Frequent Item Sets Technique
33	RLS	Replica Location Service
34	NMS	Network Monitoring Service
35	MCD	Minimum Cost and Delay
36	MMCD	Modified Minimum Cost and Delay

Algorithm 1: Coarse-grain phase algorithm
Input: *A Data Grid job (J).*
Output: *(1) ARS, (2) CTR*
Begin
1: Initialize a *min-conf* and *min-supp* /* *determined by the Broker depending on the data sets in NHF* */
2: Receive the description of Data Grid job (J).
/* *J contains name of the required files*/
3: Convert user requirements to the *QoS* metrics, such as deadline of transmission time, physical file name, file size and required cost using *Resource Requirement Interpreter*
4: An abstract resource requirement is passed to *DGRIS* unit to discover all the resources that are matching the user/application requirements with the dynamic attributes such as Bandwidth, Delay and Cost. For that, *DGRIS* contacts two core services, (1) *RLS* to get List of IP addresses of replica providers then, (2) *NMS* to get network monitoring reports.
5: Call *DM-ARD* (*in: NHF, min-conf, min-supp*, out: *ARS, CTR)* as shown in Fig. 6 to get *ARS and CTR.*
6: Send *ARS* to the *Fine-grain phase* strategy (will be explained in the next section) to get *BRS*
7: *Resource reservation interface* is used to reserve the resources of *BRS*
11: Contact Transfer Service such as *GridFTP* to transfer required file(s) from multiple providers (BRS) into computing site
11: Release resources of unselected replicas
12: Get a new job
End

Fig. 5. Pseudo steps of the Coarse-grain phase: Data Grid Mining algorithm.

Algorithm 2-a: Mining-based Associated Replicas Discovery (DM-ARD) Algorithm
Input:
(1) NHF (see Table 3) (2) min-conf (3) min-supp
Output:
(1) ARS, (2) CTR
Begin
1: Call NHF-Mapping (*in: NHF, out: LHF*) to convert NHF →*LHF*. (Fig. 8) /*convert real values of data base into logical (0,1) values (see Table 4)*/
2: Call EST (*in: LHF, min-conf, min-supp; out: FIS)* to get (ARS) Fig.9.
3: Call GRT (*in: ARS, min-conf, out: CTR*) Fig.7.
4: Call Fine-grain to extract *BRS*
End DM-ARD

Fig. 6. Pseudo steps of Data Mining-based Associated Replicas Discovery (DM-ARD) Algorithm.

Algorithm 2-b: Generating Rules Technique (GRT)
Input: *(1) ARS (2) min-conf*
Output: *CTR*
Begin
1: Call *TRT* (*in: ARS, min-conf, out: TR*) Fig 11.
2: Call *CTRT (in: TR ,min-conf, out: CTR)* to get and save it in AR^3 as shown in Fig 12.
End GRT

Fig. 7. Pseudo steps of Generating Rules Algorithm.

$\{1, 2, \ldots, k\}$ *itemsets* and save it in $ARS = \{\{L_1\}, \{L_2\}, \ldots, \{L_K\}\}$. Each iteration of the loop, say *iteration k*, generates the frequent of *k-itemsets* based on the $(k-1)$-*itemsets* generated in the

Algorithm 3: NHF-Mapping Algorithm
Input: (1) *NHF* (2) *l*: predefined window size
(3) *j*: location of the sliding window
Output: *LHF*
Begin
1: Calculate the Mean of $STT_{i,j}$:

$$MSTT_{i,j} = \sum_{k=j}^{l-1+j} (STT_{i,j}) / l$$

2: Calculate the Standard deviation of $STT_{i,j}$:

$$STDEV_{i,j} \sqrt{\sum_{k=j}^{l-1+j} STT_{i,j} (STT_{i,k} - MSTT_{i,j})^2 / l}$$

3: $Qi,j = STDEVi,j / MSTTi,j * 100$

4: Find a Coefficient Variation of Replica provider, j using: $AV_j = \sum_{i=1}^{M} (Q_{i,j}) / M$ */* AV_j is the average of $Q_{i,j}$ */*

5: Classify links into stable and unstable using the condition:
IF ($AV_j < Q_{i,j}$) then ($LV_{i,j} = 0$) Otherwise, ($LV_{i,j} = 1$)
/* *LV* represents a mapped Boolean Value of a *STT* */
6: LHF := LV;
End NHF-Mapping

Fig. 8. Pseudo steps of the NHF-Mapping Algorithm.

Table 3
Transactions table, STTs values.

STT	S_1	S_2	S_3	S_4	S_5	S	S_{29}	S_{30}
1	88	108	131	151	117	…	313	297
2	113	108	131	151	118	…	315	297
3	88.4	108	131	151	117	…	321	297
4	105	108	132	151	118	…	320	297
5	95.6	109	131	150	117	…	315	297
6	78.9	108	131	151	117	…	314	297
7	100	108	131	152	117	…	314	297
8	104	108	131	151	118	…	313	297
·	…	…	…	…	…	…	…	…
·	…	…	…	…	…	…	…	…
855	110	109	131	159	118	…	297	38.9
856	72.4	109	131	151	118	…	297	38.9
857	101	110	131	151	118	…	297	38.9
858	112	109	131	154	118	…	297	38.9
859	126	110	132	153	118	…	297	38.9
960	95.8	109	131	153	118	…	297	38.9

Table 4
Transactions table, STTs values.

STT	S_1	S_2	S_3	S_4	S_5				S_{29}	S_{30}
1	0	1	1	1	1	…	…	…	1	1
2	0	1	1	1	1	…	…	…	1	1
3	0	1	1	1	1	…	…	…	1	1
4	0	1	1	1	1	…	…	…	1	1
5	0	1	1	1	1	…	…	…	1	1
6	0	1	1	1	1	…	…	…	1	1
7	0	1	1	1	1	…	…	…	1	1
8	0	1	1	1	1	…	…	…	1	1
·	0	…	…	…	…	…	…	…	…	…
·	0	…	…	…	…	…	…	…	…	…
855	0	1	1	1	0	…	…	…	1	1
856	0	1	1	1	0	…	…	…	1	1
857	0	1	1	1	0	…	…	…	1	1
858	0	1	1	1	1	…	…	…	1	1
859	0	1	1	1	1	…	…	…	1	1
960	0	1	1	1	1	…	…	…	1	1

Table 5a
Sample of Logical History File (LHF).

Session #	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8
1	0	1	0	1	0	1	0	1
2	0	1	0	1	0	1	0	1
3	1	1	0	0	0	1	0	1
4	1	1	0	0	0	1	0	1
5	0	1	1	1	0	1	0	1
6	0	1	0	1	0	1	0	1
7	0	1	1	1	0	1	0	1
8	0	1	1	1	0	1	0	1
9	0	1	1	1	0	1	0	1
10	0	1	1	1	1	1	0	1
11	0	1	1	1	1	1	0	1
12	0	1	1	1	1	1	0	1

Table 5b
Example of Frequent Item Sets.

1-itemset $L_1 = \{\{S_3\}, \{S_4\}, \{S_6\}, \{S_8\}\}$
2-itemset $L_2 = \{\{S_3, S_4\}, \{S_3, S_6\}, \{S_3, S_8\}, \ldots, \{S_6, S_8\}\}$
3-itemset $L_3 = \{\{S_3, S_4, S_6\}, \{S_3, S_4, S_8\}, \ldots, \{S_4, S_6, S_8\}\}$
4-itemset $L_4 = \{\{S_3, S_4, S_6, S_8\}\}$

Algorithm 4: Generate Frequent Item Sets (FIS)

Input:
(1) LHF (2) min-supp

Output: *ARS*
Begin
1: *Call Apriori algorithm (in: LHF, min-supp, out: ARS) Fig.10 [21]*
2: *Return ARS*
End ART

Fig. 9. Pseudo steps of Efficient Set Technique (EST) Algorithm.

previous iteration. This loop continues until it is not possible to generate new itemsets or the number of items in an itemset exceeds the predefined maximum number of required replicas (*MR*). *Lines* 5–12 generate all the new candidate frequent *k-itemsets* out of the frequent $(k-1)$-*itemsets*. *Lines* 13–16 remove those candidate frequent *k-itemsets* that do not fulfil the minimum support requirement. In *line* 22 the algorithm returns all generated Frequent Item Sets. Table 5b, shows an example of the Frequent Item Sets generated from *LHF*. Table 5a indicates, by applying the Frequent Item Sets generating algorithm with a minimum support of *min-support* $= 50\%$, and maximum number of associated replicas $MR = 4$.

5.1.2. Transferring rules (TR)

This is the second stage of the Coarse-grain phase. It is used to generate a set of transferring rules for the AR^3. These rules will be used to select providers for identical Data Grid jobs (requested files indices are identical or closed), generating TR and then evaluating TR. The transferring rules process is explained in the following subsections:

5.1.2.1. Generate transferring rules technique (TRT):. The association rules are generated from frequent *k-itemsets* after applying *DM-ARD* and getting *ARS*. In our work these association rules are called *Transferring Rules (TR)*. Transferring rules are formed as $x \rightarrow y$, where x is a $(k-1)$-*itemset*, y is a 1-itemset and $AR = x \cup y$.

Fig. 11 shows the steps of generating the rules in the *TR* algorithm. It takes a minimum confidence (*min-conf*) as an input parameter and generates all possible rules from all itemsets. However, for each Frequent Item Set, the rules are generated as follows: one item of the Frequent Item Set becomes the consequent of the rule, and all other items become the antecedent. Thus, a

Algorithm 5: Apriori Algorithm
Input:
(1) *LHF* (2) *min-conf* (3) *min-supp*
Output: *ARS*
Begin
Initialize (1) $k = 1$, (2) C_1= *all the 1-itemsets* /*set of sets with single replica site*/
1: Read *LHF* to count the support of C_1 to determine L_1, L_1= { $\forall$*(1-itemsetes)* $\in$ C_1 and *s(1-itemset)* $\geq$ *min-supp*
2: k =2 /*k represents the pass number*/
3: Read *MR* /*Maximum number of replicas */
4: While ($L_{k-1} \neq \emptyset$ and $k+1 \leq MR$) do
5: begin
6: $C_k = \emptyset$ /* start generating candidate-itemsets*/
7: for all itemsets $l_i \in L_{k-1}$ do
8: for all itemsets $l_j \in L_{k-1}$ do
9: if ($l_i[1] = l_j[1] \wedge l_i[2] = l_j[2] \wedge \ldots \wedge l_i[k-2] = l_j[k-2] \wedge l_i[k-1] < l_j[k-1]$, *then* ($c = l_i[1], l_i[2], \ldots, l_i[k-1], l_j[k-1]$)
10: $C_k = C_k \cup \{c\}$
11: for end
12: for end /*end of generating candidate-itemsets*/
13: for all candidate itemsets $c \in C_k$ /*start pruning process*/
14: for all *{(k-1)-subsets of c }* do
15: if ($d \notin L_{k-1}$) then $C_k = C_k - \{c\}$ /*Delete c from C_k */
16: for end /*end pruning process*/
17: for all transactions $t \in T$ do
18: L_K = All candidates in C_k with minimum support
19: $k=k+1$
20: While end
21: end *ART*
22: Return $ARS = \cup_K L_k$
End Apriori

Fig. 10. Pseudo steps of Apriori Algorithm.

Algorithm 6: Generate the association Transferring Rules Technique (TRT)
Input:
(1) ARS
(2) min-conf
Output:
TR
Begin (TRT)
1: Initialize: $TR = \emptyset$ *and* $k = 2$
2: While (while $L_k \neq \emptyset$) do
3: for each itemset ($l_i \in L_k$) do
4: for each itemsets ($l_i[j] \in l_i$) do
5: if $(s(l_i) / s(l_i - l_i[j])) \geq$ *min-conf* , then ($TR = TR \cup \{l_i - l_i[j] \rightarrow l_i[j]\}$)
6: for end
7: for end
8: $k = k + 1$
9: While end
10: Return *TR*
End TR

Fig. 11. Pseudo steps of Transferring Rule Algorithm.

frequent *k-itemset* can generate at most k rules. For example, suppose $\{S_1, S_2, S_3\}$ is a frequent 3-*itemset*. It can generate at most three rules: $\{S_1, S_2\} \rightarrow S_3$, $\{S_1, S_3\} \rightarrow S_2$, and $\{S_2, S_3\} \rightarrow S_1$. After generating the rules, the rule confidences are calculated to determine if they are equal to or more than the minimum confidence. Only the rules with at least the minimum confidence are kept in the rule set called TR. For instance, the confidence rule of rule $\{S_1, S_2\} \rightarrow S_3$, can be calculated as follows: $c = s(\{S_1, S_2, S_3\})/s(\{S_1, S_2\})$. If ($c \geq$ *min-conf*), the rule holds, and it will be added to the rule set *TR*.

5.1.2.2. Generate Correlation Transferring Rules Technique (CTRT). Typically the transactions differ in the number of present items (item value = "1"). Therefore, some transactions as shown in Table 5a might be necessary to be mined using Data Mining tools, to discover the hidden information that describes the relationship among the replica's providers [31]. Each transaction gives information about co-occurring items in the transaction. Using this data one can create a co-occurrence table, telling the number of times items occurred together in the whole transactions. The co-occurrence table is useful to easily establish a simple rule shown in the following example:

Rule 1 = "*Item* 2 (*Antecedent*) *comes together with* *Item* 1 (*Consequent*) *in*10% *of all transactions*".

In Rule 1, the 10% is a measure of the number of co-occurrences of these two items in the set of transactions, which is called a *support of a rule*, σ (*Rule* 1). In other words, if the frequency of *Item* 1 occurring in the set of transactions is 10%, and that of *Item* 2, 20%, then the ratio of the number of transactions that support the Consequent part of the Rule 1 (10%) to the number of transactions that support the Antecedent part of the rule (20%) gives the confidence value of the rule (Rule 1). In this case the confidence of Rule 1 is:

$$c(Rule\ 1) = 10/20 = 0.5.$$

The confidence of the *Rule* 1 is (0.5) and is equivalent to saying that when *Item* 1 occurs in the transaction, there is a 50% chance that *Item* 2 will also occur in that transaction. The most confident rules seem to be the best ones. But a problem arises, for example, if *Item* 2 occurs more frequently in the transactions (let's say in 60% of transactions). In that case the rule might have lower confidence than the random guess. This suggests the usage of another measure called the improvement or lift ratio. That measure tells how much better a rule is at predicting the consequent than just assuming the result. Improvement is given by the following formula:

$$I(TR_k) = \frac{s(Antecedent\ (TR_k) \cup\ Consequent\ (TR_k))}{s(Antecedent\ (TR_k)) \times s(Consequent\ (TR_k))}. \quad (1)$$

So, the improvement of *Rule* 1 is:

$$I(Rule) = 0.1/0.1 \times 0.2 = 0.5.$$

When improvement is greater than 1 the rule is better than the random chance. When it is less than 1, it is worse. In our case Rule 1 is five times better than the random guess.

The algorithm in Fig. 12 shows the steps of generating correlated transferring rules. Fig. 13 shows in brief the steps of the Coarse-grain algorithm.

5.2. Fine-grain: ranking and filtering process

In the *Fine-grain* phase different functions for different purposes can be used. In our work we modified the fitness function of the previous selection strategy [5].

5.2.1. Modified Minimum Cost and Delay Policy (MMCD)

Since the transport service of Data Grid, *GridFTP* [36] has an ability to send data using multiple streams to utilize the bandwidth in the grid environment, the number of streams function can also be used in a score function. The number of streams N_s is calculated using this formula [37]:

$$Ns = Bandwidth \times RTT/windowsize. \quad (2)$$

Algorithm 7: Compute the Improvement (lift)
Input:
(1) TR (2) NR (3) LHF
Output:
CTR
Begin CTRT
1: While ($k \neq NR$) do
2: for each rule ($TR_k \in TR$) do
3: Compute $s(Antecedent(TR_k) \cup Consequent(TR_k))$ in LHF
4: Compute $s(Antecedent(TR_k))$ in LHF
5: Compute $s(Consequent(TR_k))$ in LHF
6: Use Equation (1) to measure rule's correlation for each rule TR_k
7: If ($I(Rule) \geq 1$) then, $CRT = CRT \cup \{TR_k\}$ (this indicates positive correlation, otherwise, it is negative correlation)
8: end for
9: $k = k + 1$
10: end while
11: Return *CTR*
End CTRT

Fig. 12. Pseudo algorithm of CTRT.

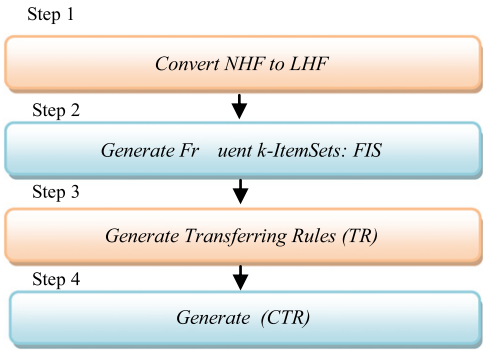


Fig. 13. Four steps of Coarse-grain phase.

As it is shown in Fig. 14, the *TCP* window size of the sender, number of streams between two sites, bit rate and *RTT* are the parameters affecting the transfer rate [38]. The *Minimize Cost and Delay (MCD)* policy [5] is modified by adding the number of streams *(Ns)* which is given in Eq. (2). The *Modified Minimize Cost and Delay* is used in the *Fine-grain* phase to get a *Best Replica Sites* (*BRS*). Fig. 15 shows the modified *MMCD* algorithm in pseudo code form. Eq. (3) is used as a scoring function in *MMCD*. The function is given as the cost per MB of a replica provider and the expected delay time for a file to finish transferring using multiple *TCP* connections. The delay time begins from the start of a transfer of a file until its completion. The minimum delay time is a *QoS* estimation based on the current available bandwidth between a potential replica provider and the computing site and a varying number of streams could be opened between them. The modified fitness function is given below:

$$SDC = -w_D(D_{ij}/Ns) - w_p(P_j \times f). \quad (3)$$

The parameters in Eq. (3) are defined as follows: *SDC* is the score based on delay and cost between two grid sites *i* and *j*,

- $\boldsymbol{D_{ij}}$: Is the estimated transfer delay time of total file between nodes *i* and *j*.
- $\boldsymbol{w_D}$: Is the weight of D_{ij}, where ($w_D > 0$).
- $\boldsymbol{N_s}$: Is the number of streams that can be opened from *i* to *j*.
- $\boldsymbol{w_p}$: Is the weight of P_j which is the estimated transfer delay between nodes *i* and *j*, where ($w_p > 0$).

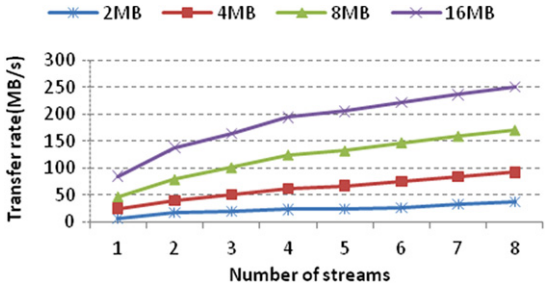


Fig. 14. Effects of varying window size and number of streams on the data transfer rate using GridFTP.

Algorithm 8: MMCD
Input:
(1) ARS, (2) f, (3)w_p*, (4)* w_D
(5) P_j *(6)*Ns_{ij}, *(7)*D_{ij} */* i represents CS and j represents replica providers*/*
Output: *BRS*
Begin
1: Read W */* determine the number of replicas providers in BRS*/*
2: for (a = 0, a < |ARS|, a++) do
3: Rank-List[a]:= SDC(ARS(a)) */* Compute SDC using Equation 3.*/*
4: end for
5: Sort-List ←Ascending Sort (Rank-List)
6: for (b = 0, b < W, b++) DO
7: BRS ← Sort-List [b]
8: Return BRS
9: end for
End MMCD.

Fig. 15. MMCD Algorithm.

- $\boldsymbol{P_j}$: Is the cost per *MB* for service provider *j*.
- $\boldsymbol{f}$: Is the file size of the requested file.

The function has placed a negative sign in front of the weights to determine the negative significance of increased cost and increased transfer delay.

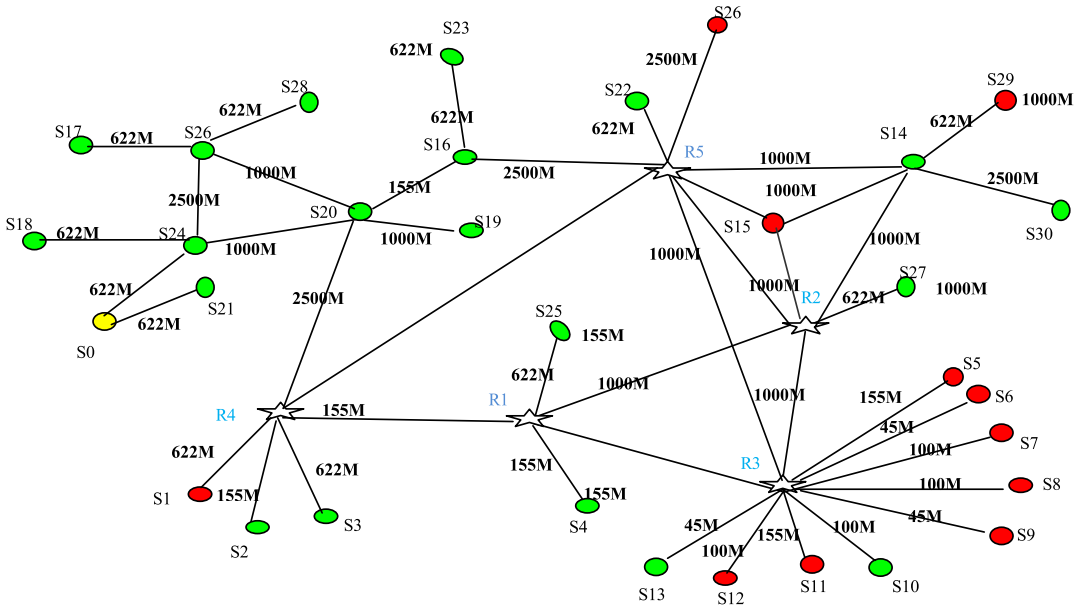
The difference between *MCD* [5] and *MMCD* can be explained in two points:

1. In *MCD*, the scoring function (Eq. (3)) is applied on all replicas whereas in *MMCD* it is applied only on the replicas in *ARS*.
2. In *MCD*, the scoring function assumed there is only one *TCP* connection that can be opened between two grid sites during calculation of the time delay of transferring the file. In *MMCD* the varying number of streams that can be opened between two sites is taken into account. Since the transport service in grid *GridFTP* has a partial file transfer feature, we can minimize the transfer time of large files by dividing the file into *W* parts and getting the best *W* number of replica sites to share the data transfer file. In Fig. 15 the *MMCD* is used to select the best *W* number of replicas from *ARS*. The parameters "*Rank-List*" and "*Sort-List*" denote the list of ranked sites using Eq. (3) and the replica sites with ascending order respectively.

6. Performance evaluation

To evaluate the 2SOB, we take up a case study which is representing a typical Data Grid environment. Then, results are compared with other methods to evaluate the selection strategy used in 2SOB.

In the Coarse-grain phase, we use authentic data from a real Data Grid environment called the LHC Computing Grid (LCG).



IP addresses of thirty sites: :S_1: 80.249.75.2, S_2: 81.91.232.2, S_3:195.24.192.36, S_4: 196.201.195.44, S_5: 193.194.185.25, S_6: 196.46.232.3, S_7 : 41.207.210.54, S_8 : 213.55.83.210, S_9 : 62.240.32.4, S_{10}: 41.203.191.3, S_{11} : 196.200.90.99, S_{12} : 82.151.64.110, S_{13}: 196.3.96.21, S_{14}: 193.95.75.236, S_{15} : 195.130.35.104, S_{16}: , 217.24.240.66, S_{17}: 161.53.160.25, S_{18}: 147.91.1.5, S_{19}: 164.8.23.111, S_{20}: 196.1.28.12, S_{21}: 93.187.162.21, S_{22}: 196.44.161.106, S_{23}: 89.250.80.3, S_{24}: 196.12.12.67, S_{25}: 202.131.0.10, S_{26}: 196.21.99.222, S_{27}: 202.38.140.115, S_{28}: 202.38.128.6, S_{29}: 137.189.27.61, S_{30}: 134.16

Fig. 16. The EU Data Grid test bed of 30 sites connected to CERN and the approximate network bandwidth.

6.1. Study case

In our simulation, we are considering the following scenario: the five files, $\{f_1, f_2, f_3, f_4, f_5\}$, each 10 GB in size are replicated and distributed on the 105 replica sites (each site holding a copy of a file is called a replica site). We are assuming a replica selection Broker to be used here, which is broadly like the one described in Section 5. Consider the situation where a grid job (J_1) is submitted to the Broker which takes up the analysis of data stored in the five files. To execute J_1 on the computing element S_0, in case the requested files are not locally available in S_0, then the files must be transferred from replica providers to S_0. To know the replica providers of these files, the replica location service is contacted by Broker to get the IP addresses of providers. In our case study, the files are found in thirty distributed sites denoted by: $\{S_1, S_2, \ldots, S_{30}\}$, as can be seen in Fig. 16.

To select a set (group) of replica providers that can be asked to concurrently send the data to S_0 we first apply the Coarse-grain then Fine-grain techniques.

The Coarse-grain method uses PingER reports and network monitoring reports, to extract a Network History File (NHF), which is explained in the next section.

Note: Transferring Rules can directly be used (Fine-grain will be ignored) when the all replica sites have identical features such as bandwidth, number of streams, etc. If replica site features are not identical, the Fine-grain phase is used to rank replica sites. Transferring Rules that have highest ranked sites will be selected.

6.2. Analysis of PingER monitoring network reports

The *PingER* report is used as a real network history file of the assumed replica providers and is collected by visiting the online distribution of historical data accessible at:

"http://www-iepm.slac.stanford.edu/cgiwrap/pingtable.pl ".

The following script is used to get a specific monitoring data at a specific date:

http://www-iepm.slac.stanford.edu/cgi-wrap/pingtable.pl?format=tsv&dataset=hep&_le=packetloss&by=by-site&size=100&tick=hourly&year=2011&month=02&day=22&from=CERN&to=WORLD&ex=none

The example above provides the hourly data from CERN to all the sites in the world for *Feb* 22, 2011. This can be imported into a text sheet which we've used in our work. The interval time can be changed to 30 min or less.

The raw data format for the data produced by the monitor contains the following fields: *(Monitor Name, Monitor Add, Remote Name, Remote Add, Time Xmt, Rcv, Minimum Average, and Maximum)*, for example (*xyz.edu*, 134:79:240:30, *abc.edu*, 134:89:240:31, 100, 10, 10, 0:255, 0:341, 0:467). Table 6 shows the meaning of the notations used in the PingER report. The number of bytes in each ping packet can be 100 or 1000.

Then for each ping response received sequence number is recorded, followed by the RTT for each ping. For example:

{0 1 2 3 4 5 6 7 8 9 0.287 0.380 0.467 0.391 0.327 0.387 0.291 0.332 0.255 0.299}.

After examining the report from the network monitoring (PingER) tool of the 30 replica providers $\{S_1, S_2, \ldots, S_{30}\}$, it has been noted that some sites are having stable network links at the same time as shown in Fig. 17. In other words, at certain time of the day, some sites have Round Trip Times with almost constant values (good latency) as it is shown in Fig. 18. Fig. 18 also illustrates the status of the link between CERN site and http://waib.gouv.bj, (81.91.232.2) as it can be seen that the stability of the link varies from time to time.

The link between the two sites was stable at the beginning of *February* 2nd 2011, then it became unstable in the middle the day, and after that it again became stable. The www.camnet.cm (195.24.192.36) site has also a stable link whereas the *univ-sba.dz*

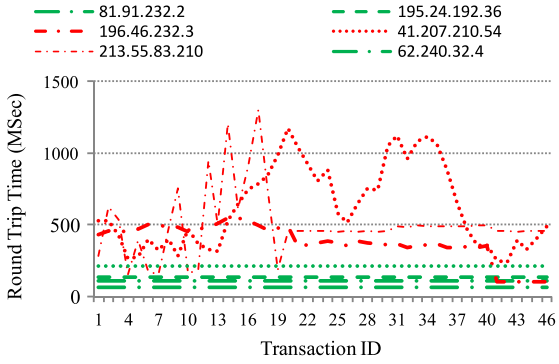


Fig. 17. Latency history of seven data providers.

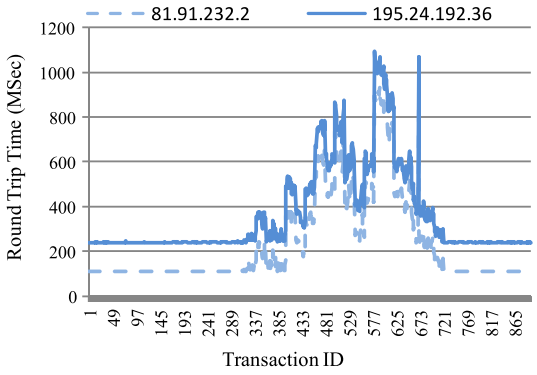


Fig. 18. RTT between http://waib.gouv.bj, www.camnet.cm and http://cern.ch.

(80.249.75.2) site has an unstable link at the same time as shown in Fig. 19. In this case, when the user sends a request to our Broker (2SOB) at the beginning of the day, both of the sites (http://waib.gouv.bj and www.camnet.cm) which had stabilized will be selected and appear in ARS after applying the Coarse-grain phase, whereas site univ-sba.dz will be discarded because it is unstable at that time as shown in Fig. 20.

6.3. Implementation and experimental results

The following steps give an overview of the obtained results:

6.3.1. The input logs files are:

1. Latency log file: In this file all round trip times between *cern.ch* 192.91.244.6 and distributed sites for the date (*February* 2nd 2011) are saved
2. The *Bandwidth (BW)*, window size and other attributes can be taken from network monitoring services such as *Iperf* [39,40].

6.3.2. The processing part

The implementation procedure for our model is done by writing a C++ program for both 2SOB phases:

6.3.2.1. Coarse-grain phase. As we mentioned, the association rules algorithm is the process of finding hidden relationships in data sets and summarizing these patterns in models [41]. The problem of mining association rules can be decomposed into two sub problems: the first is to find ARS and the second is to find the TR where both (ARS, TR) represent the output of the Coarse-grain phase. The following functions are used in the Coarse-grain algorithm to get ARS and TR.

1. Extracting RTT function: this function extracts only the RTTs values from the PingER report and saves it in a text file called the Network History File, NHF.

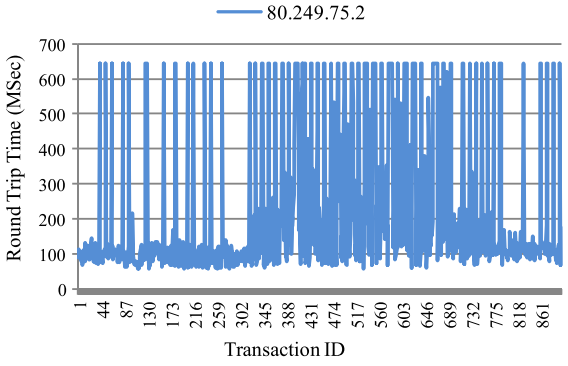


Fig. 19. RTT between univ-sba.dz and cern.ch.

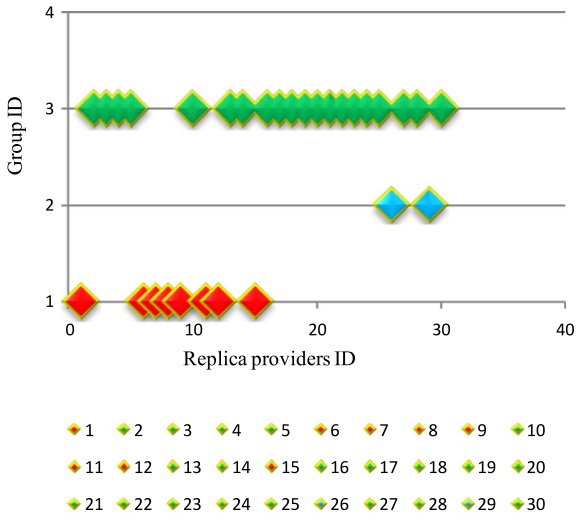


Fig. 20. Output of Coarse-grain phase.

Table 6
Notations used in PingER report.

Xmt	Number of the sent ping packets
Rcv	Number of received ping packets
Min	Minimum response time (in milliseconds)
Avg.	Average response time of the sent packets
Max	Maximum response time of the sent packets

2. Converting function: a mathematical standardization method is used to convert the real values of RTTs to logical (Boolean) values and save them in a text file called the Logical History File (LHF) as shown in Algorithm 3.
3. EST function: for discovering the Associated Replica Sites, ARS by executing the EST applying Algorithm 5.
4. Rules Transferring function: for generating Transferring Rules applying Algorithm 6.
5. Rules testing function: this function is used to compute the improvement ratio (lift ratio) of the associated transferring rules using Algorithm 7.

6.3.2.2. Fine-grain phase. A ranking function using Algorithm 8 is used in the Fine-grain phase to rank the *ARS and return BRS.*

6.3.3. The output file

The output of the proposed Broker is divided into two files. The first one is the result of the Coarse-grain phase and the second is the result of the Fine-grain phase as shown in the following:

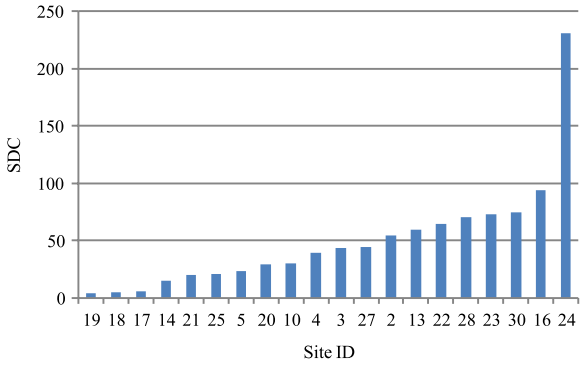


Fig. 21. Output of Fine-grain phase.

6.3.3.1. Output of phase one: Coarse-grain phase. Both stages of the Coarse-grain phase are tested.

1. Generating Associated Replicas Set (ARS): In our example when J_1 is submitted to the Two phased Service Oriented Broker at 02:00 a.m., the output of the Coarse-grain will be as shown in Fig. 20. The Coarse-grain phase separates the replica providers into three groups of providers, which are:

Group 1 : Replica sites that have unstable links, and lost packet percentage is high.
Group 2 : Replica sites that have stable links but high RTT (ARS).
Group 3 : Replica sites with good latency, and lost packet percentage is low.

In our simulation, the output of this stage is ARS which is represented by *Group* 3 and contains the following set of sites: $Group\ 3 = ARS = \{S_2, S_3, S_4, S_5, S_{10}, S_{13}, S_{14}, S_{16}, S_{17}, S_{18}, S_{19}, S_{20}, S_{21}, S_{22}, S_{23}, S_{24}, S_{25}, S_{27}, S_{28}, S_{30}\}$.

2. Generating Transferring Rules (TR): FISs are used to generate TR. The generated rules are used for making decisions (to decide which providers work concurrently to send required files at the specific time). Table 7, shows some of the transferring rules. The rules with a correlation value greater than one ($CTRT$ = "*yes*"), will be kept in the AR^3 to be used for similar jobs and file requests.

6.3.3.2. Output of phase two: Fine-grain phase. The main objective of the second phase, *Fine-grain*, is purifying the selection to get *BRS*; sites with a good throughput by applying the scoring function of Eq. (3). The replicas in ARS have different *bit rate*, *TCP sender window size* and *number of streams*. In our simulation, to test the network throughput, we assumed that the value of $W_D = W_p = 0.5$ and also all parameters of Eq. (3) are equal for all replica providers except the (Ns) and the delay (D) as shown in Fig. 21. The result of this phase is a set of sites with highest throughput and the sites having the lowest transfer time. The selected sites are saved in BRS. For instance, let us assume that two replica sites are needed to get the files. In this case the S_{18} and S_{19} are selected by the Fine-grain phase, i.e. $BRS = \{S_{18}, S_{19}\}$.

7. Comparison between Efficient Set Technique (EST) with other methods while selecting a single site

In this section we compare the performance of the Replica Broker when it uses EST (our strategy) with when it uses other strategies while selecting a single provider site. The notation in Table 8 is used in our simulation:

In our simulation we denote the time consumed during consulting the replica catalog with the logical file names to get the physical file names by α. The consumed time to probe the links between the computing site and replica sites is denoted by β. The probing time, β, is used to check availability of the file and get recent information values of the condition links such as BW. The sum of both coefficients represents the lookup time which is denoted by £, i.e., $£ = (\alpha + \beta)$. The replica selection Broker has different results while applying different selection strategies. This section is to articulate the merits/demerits between the replica Broker presented in this work and others:

7.1. Comparison between EST and traditional model (TM)

In this section we compare the total file transmit time when the replica selection Broker uses an EST strategy and when it uses TM strategies. In the traditional model which is used in Globus [2], the replica site with the maximum bandwidth or the least number of Hops (Routers) or the minimum Round Trip Time (RTT) is considered as the best replica provider. We observe that the total file transfer time using EST $T(EST)$ is calculated by:

$$T(EST) = \alpha + \beta + p(EST) \tag{4}$$

where $p(EST)$ represents the time required to execute EST and receive selection results. Further, the total file transfer time of TM is,

$$T(TM) = \alpha + \beta. \tag{5}$$

Fig. 22 shows the comparison between EST and TM using the least number of Hops and Fig. 23 shows the comparison between EST and TM using highest bandwidth. As it is observed in both figures, most of the times, the new technique adopted in 2SOB has a better performance. The reasons behind that are: EST selects the stabilized links of replica sites whereas both of traditional models do not take into account the link latency, so, the selected site is not always the best replica provider, as the link of the selected replica might be congested.

Analyzing the results: Figs. 22 and 23 show the following cases:

1. TM performance is better than EST: Under rare circumstances, especially when a single replica provider is selected, the total transfer time using TM is less than total transfer time using EST, i.e., $T(EST) > T(TM)$. This happens when $\alpha(TM) \approx \alpha(EST)$ and $\beta(TM) \approx \beta(EST)$ and finally the link of the selected replica site using a TM should be stable. The EST time is more than TM time by the time needed for executing EST, i.e. $p(EST)$.
2. Equivalent: the total transmission time for both models (EST and TM) sometimes is equal. i.e. $T(TM) \approx T(EST)$. This case happens when the time for both models (EST and TM) is equal, i.e., $\alpha(TM) = \alpha(EST)$ and also $\beta(TM) = \beta(EST)$ and sometimes, the link of the selected replica site using a TM should be nearly stable.
3. EST performance is better than TM: This case happens most of the times. The main reason is that for EST the selection decision always depends on the replicas having stable links. Stable links transmit files in less time since the number of lost packets in them is less in the stable links, i.e. $T(TM) = T(EST)$.

For clarity, let us assume the scenario of Fig. 16. A job J_1 is submitted to the computing site S_{27}. J_1 requires two files $\{f_1, f_2\}$ which are available in $\{S_3, S_5\}$. If the selection is done using the traditional model which selects the replica with the least number of Hop counts, then S_5 is selected as the best replica provider.

S_5 is selected to send the required files, because there are only two Hops (R_3, R_2) between S_{27} and S_5, whereas there are three Hops (R_1, R_2, R_4) between S_{27} and S_3. But, in this case the required time to transfer the file from S_5 is more than the required time to move the file from S_3 because the routers between S_{27} and S_3 are uncongested whereas the router R_3 between S_{27} and S_5 is congested. The same thing will happen with TM which selects the best site depending on the highest bandwidth between the two sites. But EST chooses S_3 to get the file because it has a stable link with uncongested routers.

Table 7
Sample of Transferring Rules Technique (TRT).

Rule #	Conf%	Antecedent(a)	Consequent(c)	$s(a)$	$s(c)$	$s(a \cup c)$	Lift ratio	CTRT
1	100	S_{13}, S_{15}, S_{18}	S_{14}, S_{19}	46	97	46	1.97	Yes
2	100	S_{21}, S_5, S_{18}	S_4, S_{16}	33	97	33	1.97	Yes
3	100	S_2, S_3, S_5, S_{18}	S_{14}, S_{16}	30	97	30	1.97	Yes
4	100	$S_{20}, S_{23}, S_{25}, S_{28}$	S_{23}, S_{24}, S_{16}	33	90	30	1.93	Yes
5	90.91	S_{10}, S_{24}, S_{25}	S_{22}, S_{23}, S_{17}	33	92	30	1.89	Yes
6	100	S_{18}, S_{19}	S_5, S_{17}	30	102	30	1.88	Yes
7	66.39	S_2, S_5	S_4	119	139	79	0.91	No
8	85.94	S_2, S_{16}, S_{28}	S_{30}	64	181	55	9.11	No
9	71.9	S_3, S_5, S_{16}	S_2	57	154	41	0.89	No
10	56.52	S_2, S_3, S_5, S_{17}	S_4	92	139	52	0.78	No
11	50	S_{21}, S_{17}	S_{14}	66	139	33	0.69	No
12	51.56	S_{30}, S_{16}, S_{18}	S_5	64	144	33	0.687	No

Table 8
Notations used in the simulation and results.

No.	Symbols	Description
1	$p(x)$	The execution time of x strategy.
2	$T(x)$	Total time of x selection method.
3	$\mu(x)$	Total trace time of x selection method.

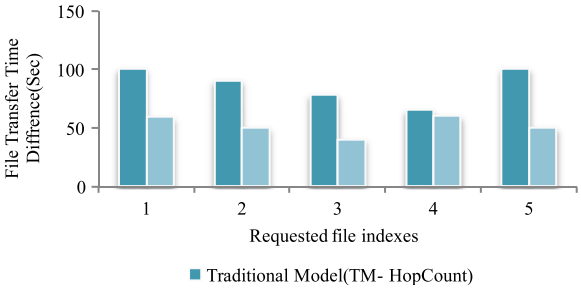


Fig. 22. Comparison between EST and TM with HopCount criterion.

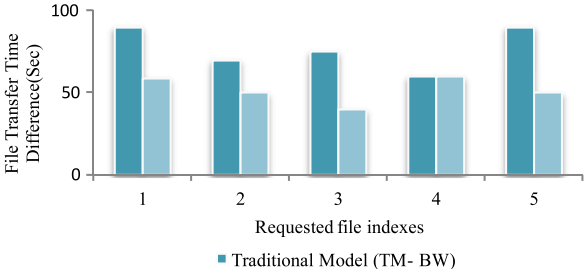


Fig. 23. Comparison between EST and TM with BW criterion.

7.2. Comparison between (EST) and neural network model (NN)

If the replica selection Broker uses a transfer time prediction model such as NN [3] or regression models, then the transferring history file (this file contains all information about previous file transfers such as: the file size, the available network bandwidth, and transfer time), is needed to be trace data and used to train the prediction model. In [3], when the history file contains at least 20 complete files transfers then the NN model starts predicting the best replica to the 21st request. The back-propagation algorithm is used to train the model. Datasets in the transferring history file are presented to the neurons of the input layer so that the output neuron can predict (estimate) the total transfer time needed to transfer file(s) from the replica providers' sites to the computing site.

The main idea is that the neural network algorithm is used to predict the current Grid transfer time for each site that holds a replica currently with respect to three factors: previous grid transfer time, current network bandwidth, and the file size. The past

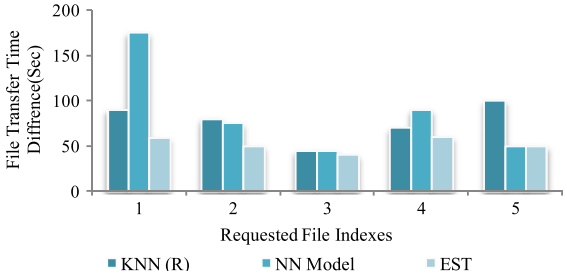


Fig. 24. Comparison between EST, NN and KNN with right selection.

data transfer throughput of sites that hold replicas can provide a reasonable approximation of the end-to-end transfer throughput. The site with the lowest predicted transfer time is chosen as the best site from which to fetch the replica. The experiments show that NN model works properly only when the bandwidth of the network links is the unique criteria for the network. However, this is not a realistic condition in the Inter-Data Grid environment. In general the latency metric is more effective than the bandwidth metric to be used in the selection strategy.

Analyzing the results: As shown in Fig. 23, both models have to contact the catalog and probe the links, so they both have a lookup time added to the total file transfer time, so:

$$T(NN) = \alpha + \beta + p(NN) \tag{6}$$

where, $p(NN)$ represents predicting time of the trained NN.
As we see in Fig. 23 the transfer time of the requested files using the neural network model is more than using EST because NN depends on two parameters which are: first, the size of requested file, and second, the bandwidth between the computing site and the replica provider site. Both of these parameters do not give a realistic picture of the dynamics of the network resources at the moment when the request for transfer was made. In Fig. 24, the following cases can be seen:

1. *Equivalent*: the total transmission time for both models (EST and NN) sometimes is equal. The reason is that both models start with lookup time and sometimes the selected replica of the NN has a stable link.
2. *Non-equivalent*: Most of the times, the EST appears to be more efficient than the NN model. The main reason is that with EST the selection decision always depends on the replicas having stable links. Stable links transmit files in less time since the number of lost packets is less. The NN model does not take the latency of the network links into account.

7.3. Comparison between (EST) and K-Nearest Neighbor (KNN) model

In this section we illustrate the difference in the efficiency of the Broker selection when using KNN and EST strategies.

The transferring history file which contains the information related to the previous jobs is needed by the KNN strategy to start its work. In other words, if the selection Broker uses the KNN model [26], then the traditional method is used for the first thirty selection tasks. The information of each selection task such as (the file indice, the destination site indice number, the requesting site indice number and the time stamp of the request) is saved as a tuple in the transfer history file. After that, the transfer history file is used to generate a decision for the request number 31 [26]. Actually,the KNN model minimizes the total file transfer time by avoiding replica lookup time ($\alpha + \beta$). That means at request number 31, the selection strategy turns from the traditional method into the KNN strategy. Then, there is no need to contact the replica catalog or to probe the network bandwidth for any new request.

In KNN, when a request for a file arrives, all previous data of the history file is considered to find a subset with K of previous file requests (K tuples) similar to the new request. Then the K similar requests are used to predict the best replica provider to the request number 31. In [26], to evaluate the KNN strategy, they compare the result of the selection between KNN and TM strategies. If they both (KNN and TM) select the same replica site, the file transfer is classified as a right classification; otherwise it is classified as a wrong classification.

7.3.1. Analysis of KNN

Since the KNN model needs at least 30 completed file transfers to begin, so the comparison is made up of two parts. The first part is related to the time of the first 30 files transferred and the second is related to the time of other file transfers starting from 31. As we mentioned, the first part of KNN uses TM and the comparison between TM and EST is done above Section 7.1. In Fig. 21, we compare the efficiency of the EST with right and wrong classifications of the KNN of the second part (starting from request number 31).

7.3.2. EST and KNN with the right classifications

Even though the KNN model predicts a right classification, the efficiency of the selected site is less than EST because the Grid environment is dynamic where user requests and network latencies vary significantly, therefore the site selected by KNN may not be the best site for replica selection under this condition. In Fig. 24, the following cases can be seen:

1. KNN performance is close to EST: Under some circumstances, especially when a single replica provider is selected to send the file, the total transfer time using KNN is close to the total transfer time using EST. i.e., $T(EST) \approx T(KNN)$. This case happens when $£(KNN) \approx £(EST)$ and the selected replica by a KNN has a stable link.
2. EST performance is better than KNN: This case happens most of the times. The main reason is that for EST the selection decision depends on the replicas which have stable links whereas the KNN does not care about the link latency. Stable links transmit files in less time since the number of lost packets are less in the stable links i.e., $T(EST) < T(KNN)$. In case of right selection the total time is,

$$T(KNN - R) = p(KNN). \tag{7}$$

7.3.3. EST and KNN with the wrong classifications

Due to the sudden change in network conditions, the KNN rule model may give wrong classifications. In [26], where KNN is used as a selection replica strategy, the simulator of the KNN replica selection model switches to the traditional model again if the KNN model gives five consecutive wrong classifications. As shown in Fig. 25, the wrong classification makes the transfer time

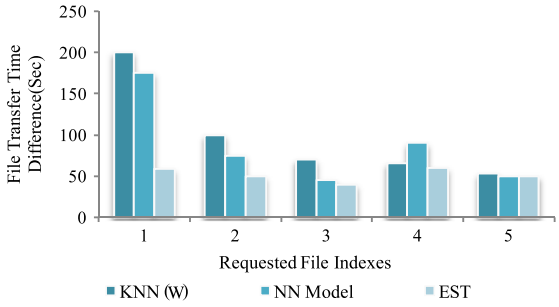


Fig. 25. Comparison between EST, NN and KNN with wrong selection.

longer than the right one because the time of wrong classification $T(KNN - W)$ comes from the summation of time spent to contact the predicted site which does not have the file any more and the replica lookup time of the traditional model as shown in the following equation:

$$T(KNN - W) = p(KNN) + (TM). \tag{8}$$

In Figs. 24 and 25 we use $K = 5$ to depict the effect of right and wrong classifications on the total file transfer time for different types of file access patterns of the file requests such as sequential and random, for example.

7.4. Comparison between (EST) and Genetic Algorithm (GA) model

In this section we illustrate the difference in the efficiency of the selection Broker when using GA and EST strategies. A *Rank based Genetic Algorithm* is used as a new technique to enhance the replica selection problem [15]. The main objective of using GA is to reduce the time of searching for an appropriate replica provider who has a close matching to the user/application request. GA is used as a clustering method. The purpose of GA is grouping replica providers into k number of clusters $\{z_1, z_2, \ldots, z_k\}$. The *Clustering Metric (CM)* that has been adopted is the sum of the Euclidean distances of the points (replicas) from their respective cluster centers. The task of GA is to search for the appropriate cluster centers such that the clustering metric CM is minimized. After clustering replica providers, *Euclidean distance equation* is used to find out all distances between users' requests and z_l, where, $l = \{1, 2, \ldots, k\}$. Then the cluster with minimum distance is selected since it contains the best provider.

There are two slots of time in the GA model. First the time of executing GA (time taken to trace GA with all possible generations), which is denoted by, $\mu(GA)$ and the second slot of time is used to run GA, which is denoted by, $p(GA)$, so the total transfer time using GA is:

$$T(GA) = £(GA) + \mu(GA) + p(GA). \tag{9}$$

GA has two options to work, which are:

1. *GA runs online selection*: in this case the GA strategy runs when the Data Grid job is submitted. In this case, the total transfer time using GA:

$$T(GA - online) = T(GA). \tag{10}$$

2. *GA runs offline selection*: in this case the GA strategy runs before submitting the Data Grid job. This assumption works properly when the user/application does not specify a constraint time for executing the job. In this case, the total transfer time using GA is,

$$T(GA - offline) = p(GA). \tag{11}$$

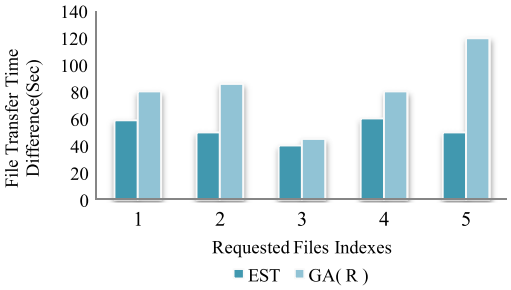


Fig. 26. Comparison between EST and GA with right selection.

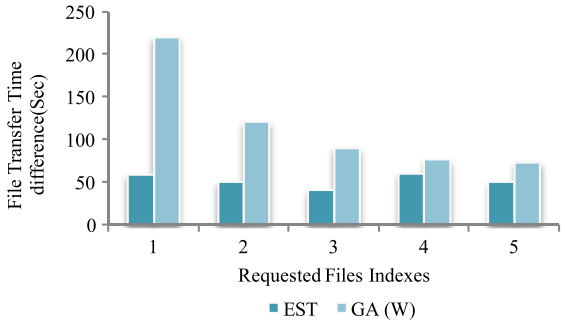


Fig. 27. Comparison between EST and GA with wrong selection.

The following subsections give an analysis of GA.

7.4.1. EST and GA with the right selection:

Even though the GA model predicts a right selection, the efficiency of the selected site is less than EST because GA needs time to get results whereas in the Grid environment some of the replica providers' attributes are dynamic such as (BW) and the network latencies vary significantly, therefore the site selected by GA might not always be the best site after some lapse of time. In Fig. 26, the following cases can be seen:

1. GA performance is close to EST: Under three special conditions, first, when the selection process is about only one provider and the second is, the work is done offline i.e., $\mu(GA) = 0$, and last, the selected provider by a GA has a stable link. Under these three conditions, the total transfer time using GA is close to the total transfer time using EST, i.e. $T(GA) \approx T(EST)$.
2. EST performance is better than GA: This case happens most of the times; EST is more efficient than GA. The main reason is that EST chooses the replicas which have stable links whereas the GA does not care about the link latency. Stable links transmit files in less time since the lost packets are less, i.e., $T(EST) < T(GA)$. In case of right selection the total time is:

$$T(GA - R) = £ + p(GA). \tag{12}$$

7.4.2. EST and GA with the wrong selection

Due to a sudden change in network conditions, the GA model may give the wrong provider which may not have a copy of the file. As shown in Fig. 27, the wrong selection makes the transferring time longer than the right one because the time of the wrong selection $T(GA - W)$ comes from the summation of time spent to contact the predicted site which does not have the file any more (the file has been deleted) and the replica lookup time of traditional model as shown in the following equation:

$$T(GA - W) = T(GA) + T(TM). \tag{13}$$

There are two reasons that make GA unsuitable for work in Grid environment.

1. GA needs a long period of time to generate a sufficient number of generations and then to give a good selection result. This is not a suitable strategy for an online Broker (jobs need to be immediately executed).
2. The dynamic nature of Grid makes GA fail since the optimizations need to be started from scratch again for any change in the provider's attribute values.

8. Comparison between EST and other methods while selecting the set of providers

To reduce the file transfer times, more than one provider may be selected to concurrently send the required files. In this section we compare the performance of EST with other replica selection strategies of the Data Grid environment while selecting more than one provider (set of replicas). As it is noted in Fig. 28, EST gives the minimum file transfer time most of the times. The reasons are:

1. Our EST technique works well with two types of replica strategies (static and dynamic) whereas most of the other models work only with a static rather than a dynamic replica strategy.
2. Our technique will select a set of sites with similar characteristics at the point of file transfer, whereas others will select one site as a best replica's site. In case these methods also apply, the same concept of having more than one site, perhaps the same result will not be achieved.
3. In previous methods, the selected replica site may or may not have the requested file since they depend on history of file requests which may be outdated information. Whereas in our method we do not have such a problem since we depend on current information from the Local Replica Catalog.
4. Some models like the traditional method depend upon the bandwidth or hop count alone which do not describe the real network condition, whereas we depend on the RTT or STT which reflects the real network conditions.
5. EST is a scalable technique, because it can deal with the file transfer problems that arise suddenly. Let us take this example: in case one of the selected providers becomes suddenly absent (leaves the grid), using the EST strategy, the Broker can immediately equip with another provider (choose another provider from the Associated Replicas Set (ARS)). Selecting a new provider does not require to re-run the EST again, compared with single replica selection methods which need to be re-run again to select extra replica sites. In brief, to get W providers using other mentioned methods, the selection strategy must be re-run W times. The re-run process adds extra time to the total file transfer time which makes the method less efficient than EST.

9. Performance and results

This section describes the performance of our proposed selection strategies in the 2SOB. The first phase, known as Coarse-grain has been applied on real network data (February, 2011) obtained from CERN [10,11]. Comparisons were made with other contemporary selection methods of different Brokers and also with various replica selection methods that were proposed in the literature. Tests have been performed in two different selection scenarios:

1. Selection of a single replica provider and,
2. Selection of multi-replica providers.

In our experiments, we have used nine jobs, each requests a single file. The sizes of files are (10, 32, 54, 27, 20, 30, 44, 50,16) GB. The files were replicated in 30 replica providers in different locations. Six selection methods (including ours) have been used in our experiments which are: (TM (BW), TM (HopCount), NN, KNN,

Table 9
Comparison between EST and others using a single replica provider.

Selection method	$Av(T)$	n (%)
TM (HopCount)	122.3	30.3
TM (BW)	113.1	24.4
KNN (C)	144.4	25.3
KNN (W)	136.1	37.1
NN	126.6	32.5
GA (C)	102.4	16.5
GA (W)	145.6	41.3
EST (proposed method)	85.40	–

Table 10
Comparison between EST and others using two replica providers.

Selection method	$Av(T)$	n (%)
TM (HopCount)	95.40	32.1
TM (BW)	102.2	36.6
KNN (C)	108.3	40.2
KNN (W)	135.4	52.1
NN	114.4	43.3
GA (C)	82.40	21.4
GA (W)	93.00	30.3
EST (Proposed Method)	64.7	–

GA, EST(proposed)). Also some selection methods such as KNN and GA have been tested with both of their possibilities:

- Correct classification (predicting) (the required data was available in selected provider), and
- Wrong classification (the required data was not available in the selected provider).

The results of $Av(T)$ (represents time of running the selection method plus time of transmission the file from the selected provider to the computing site) are tabulated according to the following steps:

1. Each job is executed using all six selection methods to select the best replica provider.
2. Run time (posterior analysis) of each selection method is recorded.
3. Transfer times from the selected providers to the computing site are recorded.
4. Total time (T) is computed for all methods separately.
5. Steps (1–3) are repeated for the other nine jobs (each job has a single file).
6. Average time $Av(T)$ is computed for each selection method.

The values resulting from the simulation are tabulated in Tables 9 and 10. Furthermore, in order to demonstrate the efficiency of the EST strategy over the others, Eq. (14) is used.

$$n = (AVUS - AVOS/AVOS) \times 100 \quad (14)$$

where, *AVOS* represents Average distance value of the other strategies and *AVUS* stands for Average distance value of our strategy. The same procedure is repeated when multi-providers are selected. The results of $Av(T)$ and n for both scenarios are formed in Tables 9 and 10.

We observed that with the five methods we compared, our proposed method outperformed all others in this phase, as our method forms connected component with a singleton element with least mean of file transfer time.

To properly analyze the results of selecting single and multiple replica providers using five methods, we draw Figs. 28 and 29. These figures show that the average time, Av(T) of the proposed strategy is less compared with other selection strategies. Moreover, to find the difference between the efficiency of our strategy and the others we draw the Fig. 28 for single replica provider selection, and Fig. 29 for multiple replica provider selection. The result

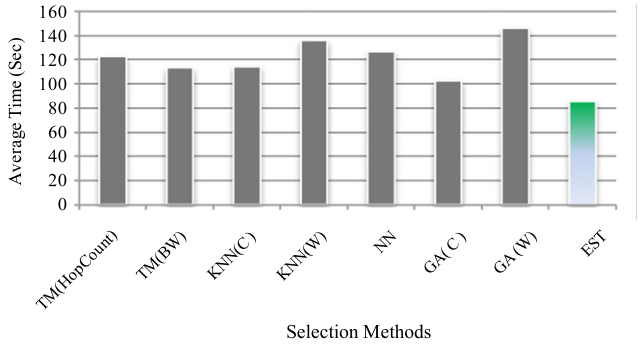


Fig. 28. Comparison between EST and other selection strategies when a single provider is selected.

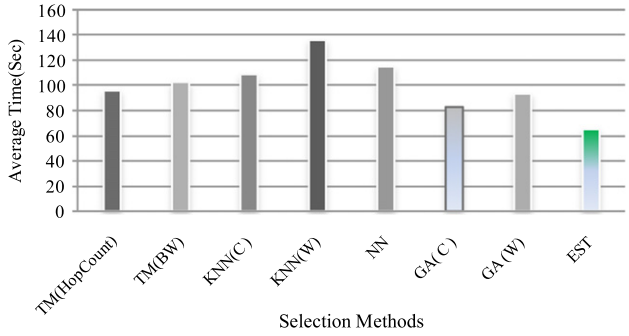


Fig. 29. Comparison between EST and other selection strategies when two providers are selected.

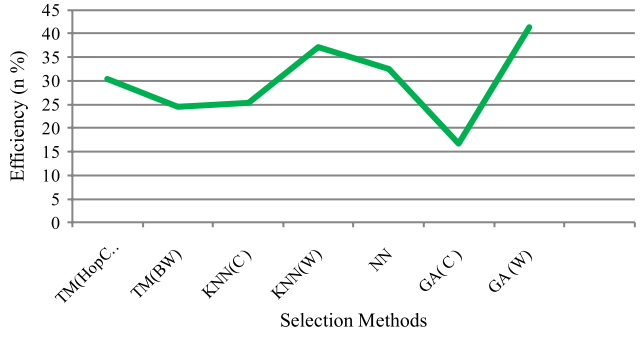


Fig. 30. Comparison between Efficiency of EST and other selection strategies when a single provider is selected.

shows that the performance of the proposed strategy EST is better compared to other selection strategies. The following are the reasons behind this:

Availability: In our proposed strategy EST there is no possibility of unavailability of the file in the selected provider since we do not avoid catalog lookup time.

Reliability: EST selects a set of providers at a time, so in case the selected provider fails another provider takes place.

Stability: EST selects providers which have stable network links, i.e., less percentage of lost packets of the required data.

Efficiency: EST selects a *set* of replica providers which are stable, reliable and available in a single run of EST. The required files are simultaneously transferred from the selected set.

Figs. 28 and 29 show that the average time of our proposed strategy EST is less when the number of selected providers increases. As we can see, the total time in the first case (when a single provider is selected) is about 80 s, and it is about 60 s in case

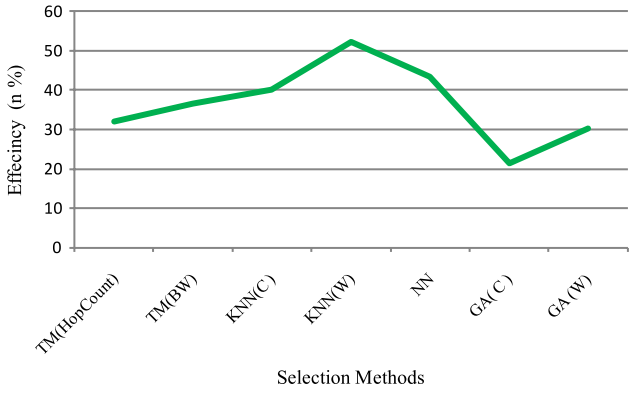


Fig. 31. Comparison between Efficiency of EST and other selection strategies when multiple providers are selected.

two providers are selected. In Figs. 30 and 31, the results show that, when the number of providers increases, the efficiency of using EST will also increase, as the effect of the associated stable links of the providers becomes apparent. That means using our proposed Broker it is possible to achieve an enhancement in the speed of executing jobs through reducing the transfer time.

10. Constraint limitations of proposed work

The work assumes the existence of Network Monitoring Reports (generally they are always available). The input files of our strategy are coming from NMS, we use RTTs instances to form the table, then we apply EST to get a set of uncongested sites.

11. Discussion and conclusions

Time and cost are the most important factors for most selection processes of the utilized grid. In this work a new data replica selection Broker has been presented with dynamic replica selection strategies. The adaptability feature addressed by the said replica selection strategy is inferred from some specific observation. This is represented by the *notion* that the basic metrics, which influence the *QoS* (that the user perceives when accessing a replica), depend directly on the application being replicated and on the clients' preferences. The enhanced replica Broker takes into account these two factors in two phases:

1. Coarse-grain phase (Associated Replicas Discovery): *Apriori* algorithm of Data Mining approach is used to extract the sites having good latency (uncongested links) in order to reduce the retransmitted data, leading to minimum transfer time.
2. Fine-grain phase (Ranking and filtering process): Here MMCD is used as a scoring function. It is adopted to use other network conditions such as bandwidth, window size of the replica site, number of streams and *RTT*.

In this work we presented a novel replica selection Broker known as 2SOB. The new Broker has the following features:

1. *Scalability*:
   2SOB is a scalable Broker for two reasons: first it is a decentralized replica brokering strategy wherein every client that requires access to a replica performs the selection process rather than a central manager performing matches against clients and replicas. Second, it uses a scalable strategy called EST.
2. *Reliability and availability*:
   In most of the previous strategies, the selected providers might or might not have the required file(s) when the transfer request was made. This situation occurs because all the previously suggested methods use outdated data for decision making (previous requests). Using our approach (2SOB), there is no possibility of non-existence of the file(s). This is because the EST depends on the current list of providers with the help of the replica location service.
3. *Transmission efficiency*:
   In the previous strategies when W providers cooperated to concurrently send parts of file(s), the strategy needed to be re-run W times as we mentioned before. The selected providers do not have a relation with each other (they do not have uncongested links at that moment when the request for transfer was made). In such a case some providers will have a stable link and some will not. The providers with unstable links will have a high percentage of retransmissions. The retransmission adds extra time to the total file transfer time. In EST the W selected providers are associated and have stable links when the request for transfer was made.
4. *Easy to deploy*:
   2SOB is an easy to deploy solution. As it is seen, the EST strategy enables organizations to improve productivity and reduce transfer costs through easy-to-deploy solutions. This is because EST could work within the existing Data Grid infrastructure for increasing the delivering operational efficiencies. Moreover outputs of EST are based only on the core services already available in the Data Grid infrastructure such as Network Monitoring Service and Replica Location Service for example.

In this work *EU Data Grid* data is used as a test data to validate the presented approach; the first phase has been applied on some real network data (*CERN*, Date: *February*. 2011). The result shows an improvement in selection using the new strategy compared with previous selection methods of different Brokers. In the future scope, it is envisaged that the presented model may use *One Way Ping Service* to get a Single Trip Time for better performance. This is because the trip from the sender to the receiver is important and affects the transfer file time whereas the trip from the receiver to the sender is not important.

2SOB can be used in a variety of Grid scientific environments, including the following projects:

Laser interferometer gravitational wave observatory (LIGO) project [42].

In this project, to meet LIGO data publication, replication and access requirements, researchers developed the Lightweight Data Replicator system [42], which integrates RLS, the GridFTP data transfer service [36] and a distributed metadata service. LDR provides rich functionality, including pull-based file replication; efficient data transfer among sites; and a validation component that verifies that files on storage systems are correctly registered in each LRC. Replication of published data sets occurs when a scheduling daemon at each site performs metadata queries to identify collections of files with specified metadata attributes. Our Broker with EST strategy can work as a selector in LDR to select a set of replica providers.

Earth System Grid (ESG) [43,44].

2SOB can work with the Earth System Grid (ESG) [44]. ESG provides an infrastructure for climate researchers that integrates computing and storage resources at five institutions. This infrastructure includes RLS servers at five sites in a fully-connected configuration that contains mappings for over one million files.

ESG, like many scientific applications, coordinates data access through a web-based portal. This portal provides an interface that allows users to request specific files or query ESG data holdings with specified metadata attributes. After a user submits a query, the portal coordinates Gridmiddleware to select a set of replica sites and to deliver the requested data. This middleware includes 2SOB (our Broker), RLS, a centralized metadata catalog, and services for data transfer and subsetting [44].

Video/Audio streaming.

2SOB can work with the Video/Audio streaming of data for organizations like the Indian National Centre for Ocean Information Systems and is one of most important applications we have to hand. We use a data turbine to place these files which carry sensor data at various locations and which can be used later for the analysis [45].

Acknowledgments

We thank *Dr. Roger Leslie Anderton Cottrell* from Stanford Linear Accelerator Center, *(SLAC)* at *Stanford University* for providing the network monitoring reports of *CERN* [10,11].

We thank Dr. Asaad Al-Salih, Ms. Yulia Sukonkina and Dr. Mahdi S. Almhanna for their help and suggestions.

References

[1] S.B. Lim, G. Fox, A. Kaplan, S. Pallickara, M.E. Pierce, GridFTP and parallel TCP support in Narada brokering, in: ICA3PP 2005, pp. 93–102.
[2] S. Vazhkudai, S. Tuecke, I. Foster, Replica selection in the globus data grid, in: First IEEE/ACM International Conference on Cluster Computing and the Grid, CCGrid 2001.
[3] R.M. Rahman, K. Barker, R. Alhajj, Replica selection strategies in data grid, Journal of Parallel and Distributed Computing 68 (12) (2008) 1561–1574.
[4] A. Abbas, Grid Computing: A Practical Guide to Technology and Applications, 2006.
[5] H. Lin, J. Abawajy, R. Buyya, Economy-based data replication broker, in: Proceedings of the 2nd IEEE Int'l Con. on E-Science and Grid Computing, E-Science 2006, IEEE CS Press, Los Alamitos, CA, USA, Amsterdam, Netherlands, 2006.
[6] A.K. Pujari, Data Mining Techniques, Universities Press, Hyderabad, 2002.
[7] R. Wolski, Dynamically forecasting network performance to support dynamic scheduling using the network weather service, in: Proc. of the 6th International Symposium on High-Performance Distributed Computing, August 1997.
[8] R.M. Almuttari, R. Wankar, A. Negi, C.R. Rao, Intelligent replica selection strategy for data grid, in: Proceeding of the 10th International Conference on Parallel and Distributed Proceeding Techniques and Applications, Las Vegas, USA, vol. 3, July 12–15 2010, pp. 95–100.
[9] J.F. Kurose, K.W. Ross, Computer Networking A Top–Down Approach Featuring the Internet, third ed, 2005.
[10] PingER. https://confluence.slac.stanford.edu/display/IEPM/PingER.
[11] Cottrell. ftp://ftp.slac.stanford.edu/users/cottrell/.
[12] S. Venugopal, R. Buyya, K. Ramamohanarao, A taxonomy of data grids for distributed data sharing, management and processing, ACM Computing Surveys 38 (1) (2006) 53. ACM Press, New York, USA.
[13] R.M. Almuttari, R. Wankar, A. Negi, C.R. Rao, Enhanced data replication broker, in: MIWAI 2011, in: LNAI, vol. 7080, Springer, Heidelberg, 2011, pp. 286–297.
[14] C.T. Yang, I. Yang, C.H. Chen, S.-Yu Wang, Implementation of a dynamic adjustment mechanism with efficient replica selection in data grid environments, in: Proc. of the 2006 ACM Symposium on Applied Computing, SAC'06, ACM, New York, NY, USA, 2006, pp. 797–804.
[15] O.A. Jadaan, W. Abdulal, M.A Hameed, A. Jabas, Enhancing data selection using genetic algorithm, in: 2010 International Conference on CICN, 2010, pp. 434–439.
[16] R.M. Almuttari, R. Wankar, A. Negi, C.R. Rao, Rough set clustering approach to replica selection in data grids (RSCDG), in: ISDA 2010, 2010, pp. 1195–1200.
[17] R.M. Almuttari, R. Wankar, A. Negi, C.R. Rao, Replica selection in data grids ufsing preconditioning of decision attributes by K-means clustering (K-RSDG), in: 2010 Second Vaagdevi International Conference on Information Technology for Real World Problems, VCON, 2010, pp. 18–23.
[18] J. Gwertzman, M. Seltzer, The case for geographical push cashing, in: Proceeding of the 5th Workshop on Hot ZTopic in Operating Systems, 1995.
[19] J. Guyton, M. Schwartz, Locating nearby copies of replicated internet servers, in: Proc. of ACM SIGCOMM'95, 1995.
[20] Cisco distributed director. http://www.cisco.com/warp/public/cc/pd/cxsr/dd/index.shtml.
[21] M. Sayal, Y. Breitbart, P. Scheuermann, R. Vingralek, Selection algorithms for replicated web servers, in: Proceeding of the Workshop on Internet Server Performance, 1998.
[22] E. Zegura, M. Ammar, Z. Fei, S. Bhattacharjee, Application-layer anycasting: a server selection architecture and use in a replicated web service, IEEE/ACM Transactions on Networking 8 (4) (2000) 455–466.
[23] R. Kavitha, I. Foster, Design and evaluation of replication strategies for a high performance data grid, in: Proc. CHEP'01, Beijing, 2001, pp. 106–118.
[24] S. Vazhkudai, J. Schopf, I. Foster, Predicting the performance of wide-area data transfers, in: 16th International PDPS, 2002.
[25] S. Vazhkudai, J. Schopf, Using regression techniques to predict large data transfers, Computing: Infrastructure and Applications, The International Journal of High Performance Computing Applications (2003).
[26] R.M. Rahman, K. Barker, R. Alhajj, Replica selection in grid environment: a data-mining approach, in: Distributed Systems and Grid Computing, DSGC, 2005, pp. 695–700.
[27] M. Litzkow, M. Livny, M. Mutka, Condor—a hunter of idle workstations, in: Proc. 8th Intl Conf. on Distributed Computing Systems, 1988, pp. 104–111.
[28] R.M. Almuttari, R. Wankar, A. Negi, C.R. Rao, M.S. Almahna, New replica selection technique for binding replica sites in Data Grids, in: EPC-IQ, Proceedings of the 2010 1st International Conference on Energy, Power and Control, Basrah, Iraq, November 2010, pp. 187–194.
[29] R.M. Almuttari, R. Wankar, A. Negi, C.R. Rao, Smart replica selection for data grids using rough set approximations (RSDG), in: CICN, 2010 International Conference on Computational Intelligence and Communication Networks, 2010, pp. 466–471.
[30] R.M. Almuttairi, M.S. Almahna, Replica selection technique for binding cheapest replica sites in data grids, in: Proceedings of International Conference of Advanced Computer Science & Information Technology, ACSIT-2012, Chennai, India, 15 July 2012, pp. 295–312.
[31] G. Williams, M. Hegland, S. Roberts, A data mining tutorial, in: IASTED Int. Conf. on Parallel and Distributed Computing and Networks, PDCN'98, 14 December 1998.
[32] R. Agrawal, T. Imielinski, A. Swami, Mining association rules between sets of items in large databases, in: Proc. ACM SIGMOD Intl. Conf. Management Data, 1993.
[33] The European datagrid project. http://www.edg.org.
[34] Mohd Farhan Md Fudzee, Jemal H. Abawajy, QoS-based adaptation service selection broker, Future Generation Computer Systems 27 (3) (2011) 256–264.
[35] OWAMP. http://www.internet2.edu/performance/owamp/.
[36] GridFTP. http://www.globus.org/toolkit/docs/latest-stable/data/gridftp/.
[37] J. Wang, L. Huang, Intelligent file transfer protocol for grid environment, in: Current Trends in High Performance Computing and Its Applications, Part II, 2005, pp. 469–476. http://dx.doi.org/10.1007/3-540-27912-1.63.
[38] H. Matsunaga, T. Isobe, T. Mashimo, H. Sakamoto, I. Ueda, Data transfer over the wide area network with large round trip time, Journal of Physics: Conference Series 219 (2010) 062056. IOP Science, 17th Int. Conf. in High Energy and Nuclear Physics, CHEP09, Japan.
[39] A. Tirumala, J. Ferguson, Iperf 1.2—the TCP/UDP bandwidth measurement tool, 2002.
[40] Iperf. http://sourceforge.net/projects/iperf.
[41] CERN. http://www.hpcwire.com/topic/storage/CERN.
[42] Lightweight Data Replicator, LIGO project. http://www.lsc-group.Phys.uwm.edu/LDR/2004.
[43] ESG Project, The earth system grid, 2005. www.earthsystemgrid.org.
[44] D. Bernholdt, S. Bharathi, D. Brown, K. Chancio, M. Chen, A. Chervenak, L. Cinquini, B. Drach, I. Foster, P. Fox, J. Garcia, C. Kesselman, R. Markel, D. Middleton, V. Nefedova, L. Pouchard, A. Shoshani, A. Sim, G. Strand, D. Williams, The earth system grid: supporting the next generation of climate modeling research, Proceedings of the IEEE 93 (2005) 485–495.
[45] A. Al-Salih, Detection of movement in video images, Ph.D. Thesis, Dept. Computer Science, Jamia Millia Islamia University, New Delhi, India, May 2010.

Rafah M. Almuttairi is currently a faculty member at the University of Babylon, Babylon, Iraq. She has completed her Ph.D. research in University of Hyderabad, India in April, 2012. She received her Master's Degree in Computer Science from Baghdad University, Iraq in October, 2003. She received her Bachelors Degree in Computer Science and Bachelor Degree in Physics from University of Babylon, Iraq in October, 2001 and October 1994 respectively. She has authored several international conference papers in the area of grid computing. Her current research interest is in the area of data grid architecture and fuzzy decision making for grid resources and replica selection.

Dr. Rajeev Wankar is working as a Associate Professor in the Department of Computer and Information Sciences at University of Hyderabad since July 2004. Before joining this University he was serving as a Lecturer in the Department of Computer Sciences of North Maharashtra University Jalgaon for ten years. He earned Ph.D. in Computer Science from the Department of Computer Science, Devi Ahilya University Indore. In 1998, the German Academic Exchange Service awarded him "Sandwich Model" fellowship. He was working in the Institut für Informatik, Freie Universität, Berlin and had collaboration with Scientists of Konrad Zuse Institut für Informationstechnik (ZIB), a Supercomputing Laboratory in Berlin, for almost two years. Currently he is working in the area of Parallel Computing, especially Parallel Algorithms design using Reconfigurable Bus System, Distributed Shared Memory Computing, Grid Computing and Multi Core Computing. He is actively participated in an International Geo-Grid activity known as GEON with San Diego Supercomputing Centre, University of California, San Diego. He served as a program committee member in many prestigious conferences such as HiPC-07, TEAA, ICDCIT, TENCON etc. He published many Journal and Conference papers and served as the Guest Editor of IJCSA's Special issue on Grid and Parallel Systems.

Prof. Atul Negi (Senior Member IEEE, Life Member IUPRAI) is a member of the IEEE SMC Techncial Committee. He has reviewed papers for several international journals like IEEE Transactions on SMC, Pattern Recognition, Pattern Recognition Letters, Image and Vision Computing, Computers and Security amongst others. He has served on the technical program committee of several conferences such as IEEE SMC, and reviewed papers for many other international conferences such as ICDAR etc. He has research interests in fields of Grid Computing, Pattern Recognition, Optical Character Recognition, and Soft Computing. He held position of Director, Prestige Institute of Engineering and Science, Indore. He worked as Scientific Officer for DRDO Project COMRADES, IISc. He worked as an Investigator for several funded projects funded by ISRO, Ministry of Communications and IT.

Prof. C. Raghavendra Rao, Completed his B. Sc and M.Sc in Statistics from Andhra University and Osmania University respectively. Ph.D. in Statistics and M.Tech (CS & Engineering) from Osmania University.

He worked as a lecturer in Statistics at Osmania University. Since 1986, he is working in the School of Mathematics and Computer/Information Sciences, University of Hyderabad. Presently he is a Professor in the Dept. of Computer and Information Sciences, University of Hyderabad.

His current research interests are Simulation & Modeling and Knowledge Discovery.

Dr Rao is a member of the Operation Research Society of Indian, Indian Mathematical Society, International Association of Engineers, Society for development of statistics, Andhra Pradesh Society for Mathematical Sciences, Indian Society for Probability and Statistics, Society for High Energy Materials, International Rough Set Society, Indian Society for Rough sets, International Rough Set Society and also a Fellow of The Institution of Electronics and Telecommunication Engineers and Society for Sciences. Dr Rao Guided 5 Ph.D.s, 40 M.Techs, 8 M.Phils. Nearly 40 Journal and 80 Proceeding Papers to his credit. He is Co-author for a book on 'Evaluation of Total Literacy Campaigns'.

Arun Agarwal completed his B.Tech (Electrical Engineering) in 1979 and Ph.D. (Computer Science) in 1989 both from IIT Delhi. He started his career as a Senior Research Assistant in IIT Delhi in 1979 and then joined University of Hyderabad in 1984, where at present he is a Professor of Department of Computer/Information Sciences.

Professor Agarwal was a Visiting Scientist at The Robotics Institute, Carnegie-Mellon University, USA and Research Associate at Sloan School of Management, Massachusetts Institute of Technology, USA. He has also visited, Monash and Melbourne University in Australia; National Center for High Performance Computing, Hsinchu, Taiwan; Chinese Academy of Sciences, Beijing, China; San Diego Supercomputing Centre USA; BioInformatics Institute in Singapore, Queensland, Australia; NECTEC, Thailand; NCSA, University of Illinois, Urbana-Champaign, USA; USM, Penang, Malaysia; KISTI, South Korea; IOIT, VAST, Hanoi, Vietnam etc.

He is on the Editorial Board of Editor, Journal of Emerging Technologies in Web Intelligence, International Journal of Pattern Recognition (RBCS); and Engineering Letters of International Association of Engineers. He is also a Fellow of Andhra Pradesh Akademi of Sciences, Fellow of IETE, Senior Member of IEEE, USA; Expert Member of AICTE to recognize new colleges to start engineering courses; Member of Board of Studies of several Universities. He was Chairman of IEEE Hyderabad Section for the years 2001 and 2002. He also received the IEEE Region 10 Outstanding Volunteer Award in 2009 in recognition of his dedications and contributions.

He has served on the technical program committee of numerous conferences in the area of Pattern Recognition and Artificial Intelligence. He has served as committee chairs of a number of these conferences. He is also on the Steering Committee of PRAGMA, Member APGrid PMA. He is a member of GARUDA project, a national initiative on Grid Computing.

His areas of interest are in Computer Vision, Image Processing, Neural Networks and Grid Computing. He has guided 9 Ph.D. thesis and more than 125 post-graduate dissertation and has published about 90 papers. He has several projects and consultancy in hand with several industry/research laboratories.

Rajkumar Buyya is Professor of Computer Science and Software Engineering; and Director of the Cloud Computing and Distributed Systems (CLOUDS) Laboratory at the University of Melbourne, Australia. He is also serving as the founding CEO of Manjrasoft, a spin-off company of the University, commercializing its innovations in Cloud Computing. He has authored 350 publications and four textbooks. He also edited several books including "Cloud Computing: Principles and Paradigms" recently published by Wiley Press, USA. He is one of the highly cited authors in computer science and software engineering worldwide (h-index $=$ 54, g-index $=$ 117, and 16 000+ citations). Software technologies for Grid and Cloud computing developed under Dr. Buyya's leadership have gained rapid acceptance and are in use at several academic institutions and commercial enterprizes in 40 countries around the world. Dr. Buyya has led the establishment and development of key community activities, including serving as foundation Chair of the IEEE Technical Committee on Scalable Computing and five IEEE/ACM conferences. The contributions and international research leadership of Dr. Buyya are recognized through the award of the "2009 IEEE Medal for Excellence in Scalable Computing" from the IEEE Computer Society, USA. Manjrasoft's Aneka Cloud technology developed under his leadership has received the "2010 Asia Pacific Frost and Sullivan New Product Innovation Award" and "2011 Telstra Innovation Challenge—People's Choice Award". For further information on Dr. Buyya, please visit his cyberhome: www.buyya.com.