Replica creation strategy based on quantum evolutionary algorithm in data grid

Tinghuai Ma a,⇑, Qiaoqiao Yan b, Wei Tian c, Donghai Guan d, Sungyoung Lee d

a Jiangsu Engineering Center of Network Monitoring, Nanjing University of Information Science & Technology, Nanjing 210044, PR China
b Hubei Meteorological Information and Technological Support Center, Wuhan 430074, PR China
c School of Computer & Software, Nanjing University of Information Science & Technology, Nanjing 210044, PR China
d Department of Computer Engineering, Kyung Hee University, Suwon 446701, South Korea

ABSTRACT

As a research branch of grid computing, data grid focuses on the management of large-scale distributed data sets. Replica management is one of the most important issues in the data grid, which can offer fast data access time, high data availability and low bandwidth consumption. Computing Intelligent Algorithm (CIA) has been proved to be effective in the solution of large-scale distributed computing problems, whereas Quantum Evolutionary Algorithm (QEA) is one of these excellent optimization algorithms and little literatures are made for its application in Data Grid Replica Management (DGRM). This paper focuses on the application of the QEA in data grid replica creation strategy. A QEA-based global replica creation strategy is proposed after reviewing the replica creation strategies. The optimization model is divided into single and multi data replica creation two parts. The representation, evaluation and constraint procedure three key technologies problems for each part are discussed in detail. The detail algorithm of QEA based replica creation is provided. The experiments were carried out with OptorSim, and the results have shown that QEA-based replica creation strategy can effectively reduce the job response time and network bandwidth consumption, comparing to Genetic Algorithms (GAs), Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO) algorithms. Especially, its performance becomes better and better with the incensement of the number of jobs. The non-parametric statistical tests are used to verify the significant of QEA.

1. Introduction

As a new research branch of grid computing, data grid [1] is designed for the application of data-intensive computing and focuses on data integration and data access with high performance. By creating one or more data replicas and assigning them to different sites, Data Grid Replica Management (DGRM) [2] technology can make grid users obtain their requiring data from local or nearby sites, which can not only shorten data response time, reduce user access latency but also save the network bandwidth, balance the server load and improve the data availability.

DGRM strategies are critical for its performance improvement. Currently, many strategies have been carried out, which mainly involve replica creation, replica location, replica selection, replica deletion and replica consistency. Replica creation strategies [3–15] decide where and when to produce replica. Replica location strategies [16,17] provide mappings from the location-independent Data Logical Name (DLN) to location-dependent Data Physical Name (DPN). Replica selection strategies [18] choose the best replica by some factors such as response time, access cost and network latency. Replica deletion strategies [19,20] remove some existing replicas to release space for new replicas. Replica consistency strategies [21,22] make all the replicas keep same when one of them is changed.

Among these strategies, replica creation is the basis of DGRM. Replica creation in data grid has been proven to be a NP problem [23]. Some researches has shown that Computational Intelligence Approaches (CIAs) is useful to the solution of this kind of problems. They are inspired by nature principles such as biological evolution, physical process and human thinking. Typical CIA algorithms mainly include Genetic Algorithms (GAs), Evolution Programming (EP), Evolution Strategies (ESs), Simulated Annealing (SA), Tabu Search (TS), Ant Colony Optimization (ACO), Immune Algorithms (IAs), Particle Swarm Optimization (PSO) and Quantum Evolutionary Algorithm (QEA). Compared with traditional optimization methods, these algorithms are robust and can be applied without recourse to domain-specific heuristics. So they have received a lot of attention in the solution of large-scale distributed problems in recent years. They have been successfully used in many fields, such as grid task scheduling [24], wireless sensor network optimization [25] and complex function optimization [26].

* Corresponding author.
E-mail address: thma@nuist.edu.cn (T. Ma).

Available online 29 January 2013

0950-7051/$ - see front matter Crown Copyright © 2013 Published by Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.knosys.2013.01.020
Compared with other fields, CIA algorithms have not usually been applied in DGRM [14,27–29], especially in replica creation [14]. Among the above algorithms, QEA [30–32] is a novel CIA algorithm developed in recent years, which combines the advantages of both evolutionary and quantum computing. By adopting qubit chromosome as a representation, the QEA can represent a linear superposition of solutions due to its probability characteristics. Compared with other CIA algorithms such as GA and SA, the QEA has rapid convergence and good global search compatibility and is a research hot spot in recent years [31–39]. However, there is no related literature made for the application of QEA in replica creation of data grid.

Based on the QEA, we proposed a method of applying QEA in data grid replica creation. The optimization model expression is provided and the procedure of replica creation is proposed.

The remainder of this paper is organized as follows. Section 2 reviews the related work. In Section 3, QEA is addressed. In Section 4, a QEA-based replica creation strategy is proposed and discussed in details. In Section 5, simulation experiments are carried out to demonstrate the performance of our proposed strategies. Conclusions and future work are drawn in Section 6.

2. Related works

2.1. Principles of DGRM

Data grid mainly consists of grid nodes and network links, which can be described as a 2-tuple ($V,E$), where $V$ is a node set and $E$ is a link set. By expanding concepts of data grid, DGRM can be abstracted as a 4-tuple ($V,E,R,O$), where $V$ and $E$ remain the same meanings. $R$ represents a replica set. It must be emphasized that master data are also treated as replica. $O$ denotes an operation set. These four elements of DGRM can be further defined as follows.

$V$ can be abstracted as a 4-tuple ($V_i,V_j,V_C,V_o$), where $V_i$ is the set of its computing elements, $V_j$ is the set of storage elements, $V_C$ is the set of jobs assigned to it, and $V_o$ is the set of replicas residing it.

$E$ can be defined by a 3-tuple ($V_i,V_j,C_i$), where $V_i$ and $V_j$ are two endpoints of edges, and $C_i$ represents the transferring cost per unit data between them, which affected by network bandwidth, disk throughput and so on.

$R$ can be represented by a 3-tuple ($logName,phyName,\text{size}$), where $logName$ and $phyName$ denote DLN and DPN, respectively, and $\text{size}$ is the amount of data.

$O$ can be described by a 5-tuple ($O_{creation},O_{location},O_{selection},O_{deletion},O_{consistency}$), which defines all the functions of DGRM in sequence of replica creation, location, selection, deletion and consistency.

The objectives of DGRM are to manipulate replicas in $R$ by the operations defined in $O$ to meet the data accessing requirements of jobs assigned to grid nodes in set $V$ as well as reduce the network traffic over $E$, shorten the job execution time, improve the data availability and increase the resource utilization. The practical operation process of DGRM can be depicted by Fig. 1.

Firstly, the job scheduling module assigns jobs to nodes in $V$, then the nodes analyze the requiring replicas of $V_j$ itself in $R$. Secondly, the nodes check whether those replicas storied at $V_i$, if exists, local processes are made to finish jobs, otherwise the operations in $O$ are used to obtain the best replicas. If necessary, some new replicas are created by $O_{selection}$. Meanwhile, $O_{consistency}$ are called to ensure that all the replicas of each data are same except that data are read-only. $O_{creation}$ maps DLN to DPN, whereas $O_{deletion}$ aims to remove the existing replicas when the available storage space of $V_j$ is inadequate for new replicas.

2.2. Replica creation strategies

Generally, replica creation strategies include two types: static and dynamic strategies. The static strategies need to obtain some correlations in advance and place replicas before jobs are executed, whereas the dynamic strategies replicate data according to the information collected at the job execution time. Compared to the static strategies, the dynamic strategies can flexibly assign replicas during jobs executions and be well adapted to different environments. However, it can also prolong the job response time as replica creation could increase the job waiting time. Both two strategies are involved in data grid replica creation. And they are often used simultaneously in the process of present strategies. For this reason, in our later discussion, we will not distinguish whether the strategies are static or dynamic. In this paper, however, the existing replica creation strategies are classified into traditional and CIA-based strategies depending on what optimization technologies they use.

2.2.1. Traditional replica creation strategies

Many replica creation strategies were proposed at the initial time when the DGRM was developed. Early in 2001, Ranganathan and Foster [3] proposed six replication Strategies. Later in 2003, Bell et al. [4] introduced the economic principles into data grid replica creation and put forward a novel replica creation strategy based on Economic Model (EM) to reduce the job execution time. Meanwhile, some EM-based replica creation strategies were implemented in OptorSim [5,6] simulator.

In 2006, Rahman et al. [7] proposed a static replica placement algorithm that placed replicas to nodes by optimizing average response time and a dynamic replica maintenance algorithm that reallocated replicas to new nodes if performance reduced over last k time periods. Tang et al. [8] put forward two replica creation strategies: Centralized Dynamic Strategy (CDR) and Distributed Dynamic Strategy (DDR), which can minimize the data access time and network load in combination with Shortest Turnaround Time (STT) scheduling algorithms.

In 2008, Wu et al. [9] discussed the problem to choose the replica placement nodes. Lei et al. [10] analyzed the data availability under the environment of file loss and bit loss when the storage capacity for replicas was constrained. Considering the user resource priority and QoS, Lin et al. [11] proposed a novel data grid replica creation strategy based on priority list which can effectively balance the replica work load.

In 2010, Sashi and Thanamani [12] proposed a novel dynamic replica creation strategy based on the popularity of files. Later in 2011, Mansouri and Dastghaibyfard [13] put forward an improved layered replica creation strategy, the experiment results showed the proposed strategy outperformed over current strategies about 14%.

2.2.2. CIA-based replica creation strategies

Compared to the traditional strategies, CIA based replica creation strategies usually adopt CIA as the main optimization algorithm. Although current research on CIA-based replica creation strategies is not so much, concerns on data grid replica creation and other aspects of DGRM are increasing.

In 2009, Naseera and Murthy [14] put forward an agent-based replica placement algorithm. Agents are deployed at each data node to determine the candidate node for the placement of replica. In 2010, Zhang et al. [15] presented a replication approach based on swarm intelligence, which was an adaptive and decentralized bottom-to-up method. Their simulation results have shown that the method performs better than no replication. And it outperforms EM for big number of jobs.
In addition, Muñoz et al. [27–29] applied ACO and PSO to replica creation and selection data grid replica selection. And the related comparison study between PSO, ACO, and other replica strategies showed that the PSO and ACO had higher performance over the traditional solutions.

3. Quantum Evolutionary Algorithm (QEA)

Inspired by the concept of quantum computing, QEA is designed with a novel qubit representation, a Q-gate as a variation operator, and an observation process. State of the art, representation and procedure of the QEA are presented in this part.

3.1. State of the art

Although QEA was proposed late compared to conventional evolutionary algorithms, much work has been done in the past few years. Early in 1996, Narayanan and Moore [30] presented the pioneering quantum-inspired genetic algorithms inspired from quantum superposition mechanism, but the traditional crossover and mutation operations were still utilized to reproduce new individual. In 2000, Han and Kim [31] proposed a novel evolutionary algorithm, called genetic quantum algorithm (GQA), and it is regarded as the first QEA by introducing the qubit chromosome and quantum gate into its algorithm. The effectiveness and the applicability of the GQA were demonstrated by experimental results on the knapsack problem [31]. Two years later, they improved GQA algorithm and renamed it as quantum-inspired evolutionary algorithm (QEA) [32]. Based on their previous work, they also proposed a two-Phase QEA scheme in 2004, which were carried out on a class of numerical and combinatorial optimization problems [33]. Shortly, Yang et al. [34] proposed a novel quantum evolutionary algorithm, where quantum mutation and quantum crossover operator were used, and some simulations were given to illustrate its efficiency and better performance than its counterpart. Simultaneously, QEA was being studied in the more complex fields and problems. In 2006, Kim et al. [35] proposed a multi-objective evolutionary algorithm (MOEA), which brought the QEA into the field of the multi-objective optimization problem. Then, Li and Wang [36] presented a hybrid quantum-inspired genetic algorithm for multi-objective flow shop scheduling based on GQA. In 2008, Vlahogiannis and Lee [37] presented an evolutionary algorithm based on quantum computation for bid-based optimal real and reactive power dispatch. Recently, in order to accelerate the convergence speed as well as avoid the prematureness, Xing et al. [38] proposed a Novel Improved Quantum Genetic Algorithm (NIQGA), where a dynamic step strategy was adopted to compute the rotation angle in quantum gate instead of the constant step. Liu et al. [39] proposed an improved INIQGA based on NIQGA. Meanwhile, by defining the conception of the angle–distance between qubits, a new Variable Angle–distance Rotation (VAR) strategy was designed to dynamically adjust the rotation angle. Base on the VAR strategy, a novel quantum evolutionary algorithm was proposed known as QEA–VAR. The experiment results have shown that the QEA–VAR has a faster convergence and better profits than other algorithms.

3.2. Representation

Definition 1. A qubit is the smallest unit of information in QEA, described by a pair of numbers ($a_1, b_1$). A qubit may be in the $|0\rangle$ state, $|1\rangle$ state, or any linear superposition of the two. The state of a qubit can be represented as $|\psi\rangle = a|0\rangle + b|1\rangle$, where $|a|^2 + |b|^2 = 1$, $|a|^2$ and $|b|^2$ give the probability that qubit will be found in the $|0\rangle$ state or $|1\rangle$ state, respectively.

Definition 2. A qubit chromosome is a string of m qubits defined by (1) where $|a|^2 + |b|^2 = 1$.

$$
\begin{bmatrix}
|a_1| & |a_2| & \cdots & |a_m| \\
|b_1| & |b_2| & \cdots & |b_m|
\end{bmatrix}
$$

A qubit chromosome can represent a linear superposition of states. For example, if there is a chromosome with three qubits and its three pairs of amplitudes is depicted by (2), then the states of system can be represented by (3). The probabilities to represent each state are the same as 1/8. By consequence, the above three qubits system have eight states information at the same time.

$$
\begin{bmatrix}
\frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2}
\end{bmatrix}
$$

$$
|\psi\rangle_{m=3} = \frac{1}{\sqrt{8}}|000\rangle + \frac{1}{\sqrt{8}}|001\rangle + \frac{1}{\sqrt{8}}|010\rangle + \frac{1}{\sqrt{8}}|011\rangle
+ \frac{1}{\sqrt{8}}|100\rangle + \frac{1}{\sqrt{8}}|101\rangle + \frac{1}{\sqrt{8}}|110\rangle + \frac{1}{\sqrt{8}}|111\rangle
$$
3.3. Procedure of QEA

The procedure of QEA is described by Fig. 3, which is similar to Genetic Algorithm (GA).

QEA maintains a population of qubit chromosomes $Q(t) = \{q_1, q_2, \ldots, q_n\}$ at generation $t$, where $n$ is the size of population, and $q_j(1 \leq j \leq n)$ is a qubit chromosome defined by (4).

$$q_j = \left[ x_1, x_2, \ldots, x_m \right]$$

(4)

In the step of initializing $Q(t)$, $x_1$ and $x_2$ of $q_j$ are all initialized with $1/\sqrt{2}$ to make all the linear superposition states represented by a qubit chromosome with the same probability:

$$|\psi_j \rangle = \sum_{i=1}^{2n} \frac{1}{\sqrt{2^n}} |x_i \rangle$$

(5)

where $x_i$ is the $i$th state represented by the binary string $(x_1, x_2, \ldots, x_m)$. In the while loop, a set of binary solution $P(t) = \{x_1, x_2, \ldots, x_n\}$ is produced by observing $Q(t)$. A binary solution $x_j(1 \leq j \leq n)$ is a binary string the same as $x_i$ which is formed by selecting each bit using the probability of corresponding qubit, either $|x|_2^2$ or $|\beta|^2$. Then each solution $x_j$ is evaluated to give its fitness and the best one is kept to make new qubit chromosomes for next generation population by the step of update $Q(t)$. In this step, new qubit chromosomes are produced by applying some appropriate quantum gates. It can make QEA have fitter states of qubit chromosomes. In this case, the key issue becomes the selection of quantum gates, which need to be designed in compliance with practical problems.

Rotation gates have been proved to be effective for the knapsack problem [31] and numerical optimization problems [40], such as

$$R(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

(6)

where $\theta$ is a rotation angle, the value of $\theta$ has an effect on the speed of convergence. But if it is too big, the solution may be diverge or have a premature convergence to a local optimum, the sign of $\theta$ determines the direction of convergence to a global optimum. And the lookup table is usually used as a strategy for the selecting of $\theta$. However, a VAR strategy [39] is adopted for its feasibility and effectiveness in this paper. The procedure of update $(q)$ is illustrated by Fig. 4.

In the procedure of Update, $(\alpha, \beta)$ of all qubits in each qubit chromosome is renovated through a rotation gate. For each qubits, the rotation angle $\theta_i$ is obtained by Eq. (4). Then the new value of $(\alpha, \beta)$ is calculated by rotation gate, such as $|\alpha_i, \beta_i \rangle = U(\theta)|x_i, \beta_i \rangle$.

The rotation angle $\theta_i$ is calculated as follows,

$$\theta_i = 1/k \times \Delta \theta_i$$

(7)

where $\Delta \theta_i$ is the angel-distance of the $i$th qubit containing the direction and step length of convergence, it can be defined as follows,

$$\Delta \theta_i = \begin{cases} \arctan(\frac{f(\beta)}{f(\alpha)}) \quad & (f(\beta) \geq (X)) \wedge (b_i \neq x_i) \\ 0 \quad & \text{other} \end{cases}$$

(8)

where $b_i$ is the $i$th bit of the best binary solution $b$, $x_i$ and $\beta_i$ are the amplitudes of $i$th qubit of $q_i$, $k$ is uniformly distributed random number ranged from 2 to 8.

4. QEA-based replica creation strategy

4.1. Motivation

It is well known that the optimum creation of replicas in large-scale data grid environment is a NP problem [23]. It is hard to find the optimum replica distributed scheme in an acceptable period of time for the traditional optimization techniques. Both the theory and application of CIA have progressed in recent years. Some efforts also have made to introduce CIA into DGRM, which provides a new way to solve the related problems. Meanwhile, as a novel CIA algorithm proposed recently, QEA outperforms well even with a small population in many applications such as 0–1 knapsack and numerical optimization problems compared to others.

Although many replica creation strategies are proposed, the replicas in those strategies are always generated during the job executing, which can lengthen the response time because longer transfer time is needed to replicate files locally. To solve this problem, we propose a global QEA-based replica creation strategy to reduce the cost of decision-making of the single node and increase the utility rate of resources, in which some replicas are pre-generated by the system control modular according to network bandwidth, storage cost, access frequency and other factors. In the
paper, QEA–VAR [39] is adopted to produce the replica creation schemes.

4.2. Model definition

For the data grid with n grid nodes and m master data files, the replica placement is to generate a certain number of replicas for m master files and distribute them to the n nodes. The optimized objective is to minimize the job response time, which mainly are influenced by the sum of storage cost \( S_c \) and transfer cost \( T_c \) of the replicas set. The replica placement models of one and more data file are discussed below to minimize the sum of storage and transfer cost, respectively.

4.2.1. Single data replica creation

In this section, taking the \( m \)th data file as an example, the replica placement model of one data file is discussed.

The storage cost of the replica set can be described by Eq. (9), where \( S_c(\mathbf{S}_m) = \sum_{i \in \mathbf{V}_m} s_i(f) \times f_m \) as the size of the data file.

\[
S_c(\mathbf{S}_m) = \sum_{i \in \mathbf{V}_m} s_i(f) \times f_m \tag{9}
\]

The transfer cost of the replica set can be calculated by Eq. (10), where \( T_c(\mathbf{S}_m) = \sum_{u \in \mathbf{V}_m, v \in \mathbf{V}_m} c(u,v) \times f_m \times f_m \) as the transfer cost per unit data between two nodes.

\[
T_c(\mathbf{S}_m) = \sum_{u \in \mathbf{V}_m, v \in \mathbf{V}_m} c(u,v) \times f_m \times f_m \tag{10}
\]

The objective of replica placement for one data file is to minimize the sum of storage cost and transfer cost, which can be described by Eq. (11), where \( \alpha \) and \( \beta \) are the weighting factors with \( \alpha + \beta = 1 \). \( S_{\mathbf{S}_m} \) as the available storage space of node \( v \), the constraint condition assures that there is enough free space for node to store the data file.

\[
\min \text{SUM}_c = \alpha \times T_c(\mathbf{S}_m) + \beta \times S_c(\mathbf{S}_m) \tag{11}
\]

subject to \( f_m \leq s_i(v \in \mathbf{V}_m) \).

4.2.2. Multi data replica creation

In this section, the replica placement model of more data files is discussed. Let \( X = \{x_{ij} | i = 1,2, \ldots , m; j = 1,2, \ldots , n\} \) denote the placement scheme of replicas, \( x_{ij} = 1 \) indicates that there is one replica of ith data file on the jth node and vice versa.

The storage cost of all the replicas can be described by Eq. (12), where \( S_c(X) = \sum_{i=1}^{m} \sum_{j=1}^{n} (x_{ij} \times s_j \times f_j) \) as the size of the mth data file.

\[
S_c(X) = \sum_{i=1}^{m} \sum_{j=1}^{n} (x_{ij} \times s_j \times f_j) \tag{12}
\]

The transfer cost of all the replicas can be calculated by Eq. (13), where \( C_t \) denotes the transfer cost per unit data between nodes \( s \) and \( j \), \( c_{ij} \) denotes the probability of node \( s \) obtaining the replica of ith data file from node \( j \). For the value of \( c_{ij} \) just as the mentioned above, if there is no direct link between nodes \( s \) and \( j \), \( c_{ij} \) is equal to the accumulation transfer cost of the shortest path between them; if \( s \) and \( j \) are the same node, \( c_{ij} = 0 \); if \( s \) and \( j \) are unreachable, \( c_{ij} = \infty \). \( T_c(X) = \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=0}^{m} (x_{kj} \times c_{kj} \times f_j \times f_j) \).

\[
T_c(X) = \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=0}^{m} (x_{kj} \times c_{kj} \times f_j \times f_j) \tag{13}
\]

Similarly, the objective of replica placement for more data file is to minimize the sum of storage cost and transfer cost, which can be described by Eq. (14), where \( s \) and \( b \) are the weighting factors with \( \alpha + \beta = 1 \). \( S_{\mathbf{S}_m} \) as the available storage space of node \( j \), the constraint condition assures that there is enough free space for node to store the data file.

\[
\min \text{SUM}_c = \alpha \times T_c(X) + \beta \times S_c(X) \tag{14}
\]

subject to \( \sum_{i=1}^{m} (x_{ij} \times f_j) \leq S_j (j = 1,2, \ldots , n) \).

4.3. Key technologies for adaptation

To solve the replica placement using QEA, some issues are needed to be considered for studying on the special problems, mainly including problem representation, solution evaluation, constraints process and convergence criterion.

4.3.1. Problem representation

It is necessary for CIA to encode the practical problems into chromosomes or solutions, and then decode the final solution to restate the problem. The most commonly used representations are binary, numeric, and symbolic. Although QEA progresses with qubit chromosomes, the binary chromosomes are the key to the presentation of practical problem for the probability characteristic of qubit. As a result, how to make the binary chromosomes associated with the replica placement becomes critical for the encoder and/or decoder. In this paper, a binary string containing \( n \) bits is used to represent one replica placement scheme of one data file, where \( n \) is set to the total number of grid nodes. If \( j \) is used to denote the index of binary string, then given value 1 represents that a replica of the file is placed at the \( j \)th node, 0 represents no replica is generated at the node.

Correspondingly, the binary chromosomes in QEA are adopted to express replica placement schemes. As mentioned above, the binary chromosomes is produced by Observing \( Q(t) \). So compared with binary chromosomes, qubits chromosomes play a more important role. To increase the diversity of solutions to obtain the optimum as great as possible, 10 qubits chromosomes are used to decide the replica placement scheme of one data file during the periods of evolutionary. As the result, the population size is set to \( 10 \times m \), where \( m \) is the total number of files. Here it is an experiential value and adopted in many QEA applications that the number of qubits chromosomes for one data replica placement schemes is set to 10. Affected by the qubit representation, QEAs do not need larger population size to assure its diversity. Finally, the best one is treated as the final replica placement scheme corresponding to the file.

4.3.2. Solution evaluation

The optimization objective of QEA-based replica placement strategy is to minimize the total cost of replica sets of \( m \) data files. To search the optimal chromosome, some evaluation functions must be designed to assign proper fitness value for each chromosome, which is very important for the whole searching process of QEA. As we have shown, the replica placement scheme of one data file is determined by 10 chromosomes, so these chromosomes must be evaluated independently to find the best scheme for the corresponding file before the comprehensive judgment is given to the replica placement schemes of \( m \) data files. In this paper, the replica placement scheme of one file is independently evaluated according to the description of (11) to find the best scheme.
for each file. After that, (14) is used to make a synthetic judgment to the best scheme of m data files. Finally, compared with the best replica placement schemes of m data files generated so far, the best one is stored as the new best schemes.

4.3.3. Constraints process

To assure the feasibility and effectiveness of replica creation schemes, as mentioned above, the total size of replicas assigned to one node cannot exceed its available storage capacity. However, it is hard to avoid violating the restraint condition because the replica creation scheme of each data is generated randomly. So the binary chromosomes must be corrected before they are evaluated for fitness, and some constraint process methods are required to correct the schemes not in accordance with the constraint conditions. The often used constraint process technologies mainly include the penalty function, decoder and repair algorithm. In this paper, the repair algorithm is adopted to correct the schemes violating the capacity constraint, which will be depicted later.

4.4. QEA-based replica creation strategy

As mentioned above, our proposed QEA-based replica creation strategy is used to create some replicas before jobs are executed. So it can be combined with some existing replica optimization strategies to reduce data response time and network bandwidth consuming. The jobs are scheduled by a system modular called resource broker according to the different job scheduling algorithms. In this paper, the resource broker selects nodes to run jobs according to the estimated time to get all files for the current job and to get all files for all jobs in the queue at the Computing Elements (CEs). If there are several replicas for users to select, that with the shortest response time has the higher priority.

4.4.1. Procedure of QEA-based replica creation strategy

As mentioned above, our proposed QEA-based replica creation strategy includes steps of parameter initialization, single data replica creation, multi data replica creation and comprehensive optimization. The procedure of QEA-based replica creation strategy is depicted by Fig. 5.

- **Parameter initialization** Parameters of r, k and m are initialized. Among them, m denotes the total number of data, r ∈ {1, 2, …, m} is a counter and k is the population size of single data replica creation, namely the replica creation scheme of one data is produced by k qubit chromosomes.

- **Single data replica creation** Single data replica creation is to generate the replica creation scheme of each data independently. To assure the constraint condition, replica is not placed into nodes whose available storage capacity is less than data size.

- **Multi data replica creation** Multi data replica creation is to make an integrated evaluation to m replica creation schemes. Repair algorithm is adopted to assure the constraint condition.

- **Comprehensive optimization** QEA-based replica creation strategy is a global replica strategy, where replicas are created before jobs are executed. In order to give a full play to our QEA-based replica creation strategy, it should be used with other replica optimization algorithm together.

4.4.2. Procedure of single data replica creation

As mentioned above, k chromosomes are used to produce one replica creation scheme to increase the population diversity in single data replica creation strategy, the procedure of which is described by Fig. 6.

In Fig. 6, Q_t denotes the quantum chromosome population used to generate the tth data replica creation scheme, where t is the evolutionary generation and q_t is a quantum chromosome with n qubits. P_t is the quantum chromosome corresponding to Q_t, where is a binary chromosome with n bits. b_i is the best-so-far binary chromosome. Q_t, q^i, P_t^i, P_t^i and b_i are illustrated by (15)–(19), respectively, where i = 1, 2, …, m, z = 1, 2, …, k and j = 1, 2, …, n

\[
Q_t = \{q_1^t, q_2^t, …, q_m^t\}
\]  

\[
q_z^j = [\frac{\beta_1^z}{\beta_1}, \frac{\beta_2^z}{\beta_2}, …, \frac{\beta_n^z}{\beta_n}]
\]  

Fig. 5. Procedure of QEA–VAR based replica creation strategy.

Fig. 6. Procedure of single data replica creation strategy.
Repeat the above steps until the cycle numbers achieve the Termination condition

- **initialize** $Q_t^i$
  
  In this step, all the quantum chromosomes of $Q_t^i$ are initialized. Concretely, $x_i$ and $p_i^t$ of $q_i^t$ are all initialized with $\frac{1}{\sqrt{2}}$ to make all the linear superposition states represented by a qubit chromosome with the same probability.

- **oberving** $Q_t^i$
  
  This step makes binary population $P_t^i$ by observing $Q_t^i$. $p_i^t$ is a binary string, which is formed by selecting each bit using the probability of corresponding qubit, either $|x|^2$ or $|p|^2$, $|x|^2$ is used in our implementation. The length of $p_i^t$ is $n$, 1 means to place one replica of the ith data in the corresponding node, 0 means not to place one replica of the ith data in the corresponding node.

- **evaluate** $P_t^i$
  
  The fitness of binary chromosomes in $P_t^i$ is calculated according to (11). Before making the fitness, free space for the site should be checked to store the replica of ith data. That is, the constraint condition $f_j \leq s_j$ should be satisfied. If the node has less free space, the corresponding bit is set to 0. The best chromosome $x_i = \{x_1, x_2, \ldots , x_j, \ldots , x_n\}$ in $P_t^i$ is compared with $b_i$, if $x_i$ is superior to $b_i$, $b_i$ is replaced with $x_i$, namely $b_i = x_i$.

- **update** $Q_t^i$
  
  The probability amplitude of each quantum chromosome in $Q_t^i$ is updated using quantum rotation gate to make according to the corresponding binary chromosome $b_i$ and which makes all the chromosomes trend to $b_i$. The rotation angle is obtained using VAR strategy.

Here, the step of update $Q_t^i$ as follows. It assumes that the node number is 5, namely $n = 5, b_i = \{0, 1, 0, 1, 0.1\}, p_i^t = \{1, 0, 1, 0, 1\}$. $q_i^t$ is the quantum chromosome corresponding to $p_i^t$, depicted by (20). The new probability amplitude of $q_i^t$ is calculated by (21) where $\theta_i^t$ is the rotation angle, which can be calculated by (22), where $\Delta \theta_{p_i^t}$ is obtained by (23).

\[
q_i^t = \left[\begin{array}{c}
x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 
\end{array}\right] \\
\left[\begin{array}{c}
p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 
\end{array}\right]
\]

\[
\left[\begin{array}{c}
x_i^t \\ p_i^t
\end{array}\right] = \left[\begin{array}{c}
\cos (\theta_i^t) - \sin (\theta_i^t) \\ \sin (\theta_i^t) \cos (\theta_i^t)
\end{array}\right] \times \left[\begin{array}{c}
x_i \\ p_i
\end{array}\right]
\]

\[
\theta_i^t = \frac{1}{k} \times \Delta \theta_{p_i^t},
\]

\[
\Delta \theta_{p_i^t} = \begin{cases} 
-\arctan \frac{\pi}{p_i} & \text{if } (f(b_i) \geq (p_i)) \land (b_i = 0) \land (p_i = 1) \\
\pi - \arctan \frac{\pi}{p_i} & \text{if } (f(b_i) \geq (p_i)) \land (b_i = 1) \land (p_i = 0) \\
0 & \text{other}
\end{cases}
\]

**Termination condition**

Repeat the above steps until the cycle numbers achieve the maximum evolution generation, which are set in advance.

### 4.4.3. Procedure of multi data replica creation strategy

As mentioned above, the replica creation scheme of one data is produced by k chromosomes. That is, the replica creation schemes of m data need $k \times m$ chromosomes. The procedure of a multi data replica creation strategy is described by Fig. 7.

As mentioned above, the replica creation scheme of one data is produced by k chromosomes. In other words, the replica creation schemes of m data need $k \times m$ chromosomes. Let $Q(t) = \{Q_1^t, Q_2^t, \ldots , Q_m^t\}$ denotes the total quantum population composed of m sub-populations, $P(t) = \{P_1, P_2, \ldots , P_m\}$ denote the total binary population composed of m sub-populations, and $B(t) = \{b_1, b_2, \ldots , b_m\}$ denotes m best-so-far binary chromosomes. The procedure of multi data replica creation strategy is described by Fig. 7.

To execute the multi data replica creation strategy, the k replica creation schemes of single data are produced according to single data replica creation strategy first. That is, each sub-population of $Q(t)$ independently evolves into $P(t)$, and each sub-population of $P(t)$ is evaluated to form the best-so-far chromosomes $B(t)$, respectively. Next, the chromosomes in $B(t)$ are evaluated comprehensively.

Similar to the single data replica creation strategy, before making the evaluation, it should assure that there is enough free space for nodes to store the replicas assigned to them. The repair algorithm is to solve this problem, which is depicted by Fig. 8. If necessary, some bits of binary chromosomes will be revised from 1 to 0.

To assure that replicas can be distributed into each node uniformly, the processing order of data replicas is generated randomly in repair algorithm. Meanwhile, it must be noted that the master files cannot be deleted during the period of repairing to guarantee that it exists at least one replica for each data in data grid. After finishing the repair algorithm, the fitness of chromosomes in $B(t)$ is calculated by (6) and save the corrected $B(t)$.

Finally, repeat the above steps until the termination condition is satisfied. $B(t)$ is output and treated as the final replica creation schemes.

### 5. Experiments and analyses

To evaluate our approach, we extended the OptorSim toolkit to incorporate the functionality of QEA-based replica placement strategy. The OptorSim [4,5] simulates a data grid environment for researchers to study the effectiveness of replica optimization algorithms.

#### 5.1. Experiments settings

**5.1.1. Experimental settings**

The simulation experiments are carried out on a CMS test bed provided by OptorSim self. The grid topology and bandwidth between grid nodes are shown in Fig. 9. The grid comprises 20 nodes and 8 routers located in the USA and Europe. The node CERN and FNAL have a capacity of 100 GB each and both of them have no CE. An original copy of each file is stored at one of them. Other nodes have one computing element and initially empty storage of capacity 50 GB.

**5.1.2. Algorithm verification**

To demonstrate the performance of the QEA–VAR based data grid replica creation strategy, No Replica (NR) and Economic Model (EM) based dynamic replication strategies provided by OptorSim itself are chosen as assistant algorithms. NR never replicates any files during both job submission and execution. However, data are sold by Storage Elements (SEs) to Computing Elements (CEs) or other SE in EM. A file is purchased by CE for running a job and by SE to make an investment. The purchasing action of a CE means the CE gets a replica of the file. CE tries to minimize the file purchase cost, while SE attempts to maximize their profits. EM is a dynamic replication strategy, which replicates files during the job execution.
On the basis of these two algorithms, two groups of experiments are carried out. Meanwhile, GA, ACO and PSO are treated as comparison algorithms. In order to facilitate the description, GA, ACO, PSO and QEA are used to denote GA, ACO, PSO and QEA–VAR based replica creation strategy in each group experiment, respectively. Compared to NR and EM, these algorithms always generate some replicas before job execution using their own strategies.

5.2. Results and analysis

5.2.1. Evaluation criteria

Mean Job Time (MJT), Effective Network Usage (ENU) and Storage Elements Usage (SEU) are taken into account to evaluate the performance of replication strategies:

- **MJT**: The smaller value indicates the shorter responding time, defined by (24), where \( t_i \) is the time of the \( i \)th job running, \( n \) as the total number of jobs.

\[
\text{MJT} = \frac{\sum_{i=1}^{n} t_i}{n} \tag{24}
\]

- **ENU**: The lower value indicates that strategies are better at putting data in the proper nodes, defined by (25).

\[
\text{ENU} = \frac{N_{\text{remote file access}} + N_{\text{file replicates}}}{N_{\text{remote file access}} + N_{\text{local file access}}} \tag{25}
\]

- **SEU**: The larger value indicates more storage resources to be resolved, defined by (26).

\[
\text{SEU} = \frac{\text{Filled space available}}{\text{Space}} \tag{26}
\]

5.2.2. Results and analysis

Before discussing the results, some parameter settings are stated here. In each group experiment, totally 97 files are used and the single file size is 1 GB. The files are accessed according to the Zipf-like distribution. The simulations are run with the number of jobs being incremented from 100 to 500. All the experiment results are averaged over 30 runs. The selection and mutation probabilities for the GA-based data grid replica creation strategy are 0.65 and 0.05, respectively. For the PSO-based replica creation strategy, the inertia factor is 0.5, the two acceleration constants are equal to 2. For the ACO-based replica creation strategy, the importance of pheromone and importance of resource innate attributes are equal to 0.5, and the permanence of pheromone is 0.8. Besides, the placement scheme of one file is produced by 100 individuals for GA-based and PSO-based replica creation strategy. That is to say, their population size is 10 times as large as that of the QEA–VAR based data grid replica creation strategy. The results are listed in Tables 1–3.

5.2.2.1. First group experiment analysis. In first group experiment, NR is treated as assistant algorithms. MJTs of GA, ACO, PSO and QEA are illustrated for the different number of jobs from 100 to 500 in Fig. 10. Among these four algorithms, it is clear that MJTs of GA and ACO are larger than PSO and QEA, which shows that the placement schemes produced by PSO and QEA data grid replica creation strategies are superior to that of GA and ACO. Meanwhile, MJTs of PSO is little smaller than that of QEA for 100 and 200 jobs. And when the number of jobs is up to 300, the difference nearly disappears. However, MJTs of QEA are smaller 9.37% and 11.45% smaller than that of PSO for 400 and 500 jobs, respectively, which proves, the adaptability of QEA–VAR based replica creation strategies.
strategies is better than PSO. Besides, the running time of QEA itself is shorter than others for its smaller population size.

To make a further study, ENUs of GA, ACO, PSO and QEA are depicted by Fig. 11. It is clear that ENU of QEA is lowest followed by PSO, ACO, and GA while the number of jobs reaches to 300 although the value of QEA is little larger than PSO for 100 and 200 jobs. So we can see that QEA based data grid replica creation strategy can put replicas to suitable nodes and reduce the consumption of network bandwidth effectively, especially for the large scale problems. However, SEU of QEA is largest followed by PSO, ACO and GA, successively, described by Fig. 12, which

![Fig. 9. The topology of CMS.](image-url)

### Table 1

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>NR group</th>
<th>EM group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job num.</td>
<td>100</td>
<td>200</td>
</tr>
<tr>
<td>GA</td>
<td>2524</td>
<td>4655</td>
</tr>
<tr>
<td>ACO</td>
<td>2200</td>
<td>3979</td>
</tr>
<tr>
<td>PSO</td>
<td>1702</td>
<td>3100</td>
</tr>
<tr>
<td>QEA</td>
<td>1722</td>
<td>3127</td>
</tr>
</tbody>
</table>

![Fig. 10. MJT of NR, CgaNR and QeaNR.](image-url)

### Table 2

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>NR group</th>
<th>EM group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job num.</td>
<td>100 (%)</td>
<td>200 (%)</td>
</tr>
<tr>
<td>GA</td>
<td>42.27</td>
<td>46.95</td>
</tr>
<tr>
<td>ACO</td>
<td>63.25</td>
<td>64.30</td>
</tr>
<tr>
<td>PSO</td>
<td>65.18</td>
<td>65.21</td>
</tr>
<tr>
<td>QEA</td>
<td>65.62</td>
<td>65.11</td>
</tr>
</tbody>
</table>
SEU of these four algorithms are also depicted by Fig. 15. Similar proper nodes compared with others in the different circumstance. VAR based data grid replica creation strategy can put replicas at the great majority, which further demonstrates that the QEA–the first group experiment ENU of QEA is still lower than PSO for the different number of jobs from 100 to 500 are illustrated in Fig. 13. It is clear that MJTs of GA, ACO, PSO and QEA here are shorter than that in the first group experiment. The main reason is that EM is a dynamic replica strategy and replicates files during the job execution. However, PSO and QEA still outperform GA and ACO. Especially, MJT of QEA is faster 17.41% and 17.76% faster than that of PSO for 400 and 500 jobs, respectively. That further demonstrates that the QEA–VAR based data grid replica creation strategy can shorten the job execution time.

Meanwhile, ENUs of GA, ACO, PSO and QEA for the different number of jobs from 100 to 500 are also illustrated by Fig. 14. The value of GA and ACO is larger than PSO and QEA. Similar with the first group experiment ENU of QEA is still lower than PSO for the great majority, which further demonstrates that the QEA–VAR based data grid replica creation strategy can put replicas at proper nodes compared with others in the different circumstance. SEU of these four algorithms are also depicted by Fig. 15. Similar results are obtained compared to the first group experiment.

Conclusively, the QEA–VAR based data grid replica creation strategy can significantly shorten the response time of jobs and reduce the network consumption. Its optimal performance is superior to that GA, ACO, PSO based data grid replica creation strategy both in speed and results. Especially, its performance becomes better and better with the incensement of the number of jobs.

5.2.3. Statistical test

As the NR, EM models lead different results, the statistical analysis will consider two groups performance measures, separately. From Tables 1–3, it can be seen that MJT obtained for each algorithm is the most important classic performance. So it is being chosen for statistical analysis. This case corresponds to a multiple-problem analysis, so the employment of non-parametric statistical tests is preferable to a parametric one [41].

In statistical analysis, we want to check whether the results of the algorithms are rather significant for considering them in different jobs. As we proposed the QEA algorithm, which has lowest MJT in the comparison will be studied carefully. The p-values on each comparison will be computed to determine the results QEA offers are better than the ones offered by the rest of algorithms.

Table 4 shows the results of applying Friedman’s order to see whether there are global differences in the results. Given that the p-values of Friedman are lower than the level of significance considered $\alpha = 0.05$, there are significant differences among the observed results in the NR group and EM group.

From Table 5, the QEA algorithm outperforms the GA, ACO with a level of significance $= 0.05$, because the all p-values of QEA vs. GA and QEA vs. ACO are lower than. Here we cannot declare that QEA obtains a significantly better performance than PSO algorithms, due to the fact that p-values are 0.225 and 0.5 corresponding to NR and EM. However, it is obvious that the QEA algorithm outperforms the PSO in the large scale problems, which can be seen from the Figs. 11 and 13.
Acknowledgement

This work was supported in part by National Science Foundation of China (No. 61173143), China Postdoctoral Science Foundation (No. 2012M511783), Jiangsu Qing Lan Project and Special Public Sector Research Program of China (No. GYHY201206030) and was also supported by PAPD.

References


