Received <day> <Month>, <year>; | Revised <day> <Month>, <year>; | Accepted <day> <Month>, <year>

DOI: xxx/xxxx

SPECIAL ISSUE PAPER

A New Replica Placement Mechanism for Mobile Media Streaming in Edge Computing†

Yayuan Tang[1,2] | Hao Wang[3] | Kehua Guo*[1] | Tao Luo[1] | Tao Chi[4]

[1]School of Computer Science and Engineering, Central South University, Changsha, China
[2]School of Electronics and Information Engineering, Hunan University of Science and Engineeringme, Yongzhou, China
[3]Department of Computer Science, Norwegian University of Science and Technology, GjÃ¿vik, Norway
[4]Key Laboratory of Fisheries Information, Ministry of Agriculture, Shanghai Ocean University, Shanghai, China

Correspondence
*Kehua Guo, School of Computer Science and Engineering, Central South University, Changsha, China. Email: guokehua@csu.edu.cn

Abstract

With the advent of the Internet of Things era, cloud computing platforms will face the challenges of massive equipment requirements for access, massive data, insufficient bandwidth and high power consumption. Edge computing, as a new technology, makes it possible to stream media over the edge network. However, many problems related to file sharing among edge nodes and to the files provided by cloud servers exist. We propose a novel replica placement strategy for mobile media streaming in edge computing (RPME) to address the aforementioned problem. Firstly, we introduce a multi-level replica placement model in the RPME. In the RPME, we acquire the user information and the user-item rating matrix when the user requests the media data. At the server, we cluster the users according the user information, update the user-item rating matrix, and then generate a replica recommendation sequence. For when the server submits the replica recommendation sequence to the edge node considering the constraints of the edge node storage capacity and the limitations on the service capacity of the requested replica, we propose an effective replica placement mechanism in the RPME. To show the benefits of the RPME, we also present several experiments to prove its validity.

KEYWORDS:
Resource replica; Placement strategy; Edge computing; Load balancing; *Time delay*

1 | INTRODUCTION

With the full speed advances of mobile computing, there are increasingly more scenarios and needs for users to access Internet resources in a mobile environment. People will turn to the ubiquitous mobile computing era from the personal computer era. Meanwhile, many wireless access networks, such as GPRS , Wi-Fi and other cellular mobile networks, offer mobile users with the probability of ubiquitous and heterogeneous wireless access networks. Consequently, the amount of media data on the Internet has grown dramatically [1,2]. Cisco predicts that 75% of all global mobile data traffic will be video in 2020 [3].

Worst of all, while a large number of users can access the same resource simultaneously, the trunk network traffic and access latency will significantly increase, which can cause system bottlenecks [4]. People have widely used conventional CDN and P2P technology in mobile environments. A hybrid P2P-CDN system for media streaming was established to develop various mobile services [5]. Due to the strong dynamic mobility and topological changes in the mobile network, the system is poor and unstable. The degradation of the network utility in a dynamic network was studied using NFV [6]. Wang HT presented a selection algorithm for the super node in the MP2P-CDN, which improved the productivity of a mobile content distribution network [7]. Replica placement technology is usually used to mitigate bottlenecks and improve user experiences. Existing studies on replica placement in HCDNs [8] are limited to the content replication and adopt a non-cooperative pull-based approach. Many solutions have been

† This is an example for title footnote.
[0]**Abbreviations:** ANA, anti-nuclear antibodies; APC, antigen-presenting cells; IRF, interferon regulatory factor

discovered to reduce the number of deployed servers [9]. However, a comprehensive scheme that is scalable, reliable, responsive, and cost-effective is still an illusory goal [10].

Increasingly more users tend to use video streaming, web browsing, social networks and online games, thus forcing vendors to develop new service technologies to provide an acceptable quality of service (QoS) [11,12]. The appearance of edge computing provides a new dawn for this predicament. Edge computing is an enabling technology that allows for computing on network edges, including downstream data representing cloud services and upstream data representing IoT services. Edge computing will have a huge influence on our society, similar to cloud computing [13]. Edge computing has an important feature that each node can be calculated and stored so that the transparent learning (TL) framework can transfer the tasks of training models to servers and edge IoT equipment [14]. In addition, deploying edge servers with storage and computing capabilities can reduce the number of edge nodes and improve the throughput between nodes and edge nodes [15]. The calculation and storage of each node are fully utilized.

In this paper, we present a novel replica placement mechanism for mobile media streaming, which is an effective framework that facilitates QoS performance in edge computing. The main contributions of this paper are summarized as follows.

1) For the first time, we introduce a new replica placement model in the RPME.

2) In the RPME, we present optimal solutions to deal with the replicas based on the block-level storage and define a linear sequence to fetch data from the nearest sever. We further design a set of practical solutions for dynamically placing the replicas in order to improve the performance in edge computing.

3) To understand the performance of the RPME, extensive simulation experiments have been carried out. The experimental results demonstrate that the proposed RPME effectively deploys the resource replica to balance the loads of the edge nodes and provide users with satisfactory streaming latency.

The rest of this paper is organized as follows. The related work is concise and to the point and we compare it with similar problems in section 2. Section 3 elaborates on the architecture of the RPME and the formal definition of the replica placement problem. In section 4, as the solution to the problem, the effective replica placement policy is expounded. In section 5, the proposed strategy is implemented, some experiments are carried out, and the simulation results are compared with other strategies. Finally, we summarize the paper and discuss our future work in section 6.

2 | RELATED WORK

Over the past ten years, researchers have done much research on replica placement in CDNs [16-18]. The used replica placement algorithm (RPA) was a minimum p-median problem, which was an NP-hard problem [19]. The most representative RPAs include Greedy [16], Hot Spot [17] and Random [18]. The greedy algorithm iterates over the server to reduce the replication costs. The Hot Spot algorithm determines the location of the replica according to the number of user requests. The random algorithm randomly selects some locations at which to place replicas on the server. Because static RPAs cannot accommodate rapidly changing scenes, the new dynamic RPA method was proposed based on the number of data flows by network nodes [20]. To guarantee the QoS requirements, the object replication algorithm was designed [21]. In addition, a robust replication layout is proposed to increase performance under the nondeterminacy of aleatoric server failures [22].

Since nodes can connect or exit the network at all times in a P2P network, the replica placement solution in the CDN network cannot be used directly in a P2P network. By analysing the relation among the node connectivity, data popularity and query success rate, reference [23] proposed a dynamic replication method in view of the topological structure and provided a best distributed model for node connectivity. A hybrid genetic algorithm was proposed for scheduling local ordered tasks [24]. However, the model did not consider the interactions between nodes when content replicas were spread over diverse nodes in the network, and thus an evolutionary game theory model of a P2P network was proposed [25]. Although researchers have made many improvements, the above algorithm cannot be directly applied to mixed CDN-P2P networks because CDN-P2P networks have the advantages of CDN networks and P2P networks.

Considering the characteristics of the CDN-P2P, an economical heuristic solution was proposed to resolve the replica placement problem of multicast tree construction, which was proved to be np-hard [26]. To solve this problem, RPAs were considered to be a constrained p-median problem in terms of the transmission and storage costs for the mixed CDN-P2P [24]. Nevertheless, any replication algorithm is insufficient due to the high transient nature of wireless and mobile devices. Reference [27] constructed a novel real-time streaming scheme in which a new hybrid CDN-P2P strategy established buffers in CDN servers and P2P networks. Due to the dramatic increase in mobile multimedia data, the user experience has plummeted. Reference [28] addressed the replica placement policies that have been applied in a peer-to-peer video content delivery network based on smart routers to improve the user QoS in recent years. To reduce the initial and playback time delays, the researchers also proposed a hybrid P2P-CDN system for media streaming in mobile environments [29].

Existing studies on replica placement usually considered normal networks when improving the average performance indicator, such as the response delay [30], load balancing [31], communication costs and storage overhead [32], but they overlook the needs of users and are unable to

offer individualized web services. A QoS-aware replica placement algorithm satisfies the QoS of all users by improving the handling of the demands of users [33]. A cognition-based predictive model in content popularization that combined the contextual information and user interests placed the replicas focusing on user demand [30] in a way that relatively increased user satisfaction. Nevertheless, the provided service is still short on quality. In this work, we propose a personalized recommendation replica placement mechanism, which is called the personalized recommendation replica placement. This solution can improve the service ability of the server and can satisfy the individual user needs with respect to content.

3 | SYSTEM MODEL AND PROBLEM DESCRIPTION

This section describes the process of transferring media streaming from the server to the client in the system model and formalizes the media replica placement problem.

3.1 | The logical view of RPME

Edge computing adheres to the C/S architecture and cloud computing. As the quantity of clients increases, the system performance might be degraded because the large number of I/O requests sent to the server may surpass the maximum I/O throughput of the server's external storage devices, such as the hard disks. This situation may lead to slow client responses and reduced user experiences. Therefore, reasonable replica placement will greatly improve system performance. In edge computing, it can take advantage of the computing power of each edgeNode, which is connected between the server and the client. Simultaneously, we can also place replicas in these edgeNodes to increase the systemâĂŹs performance.

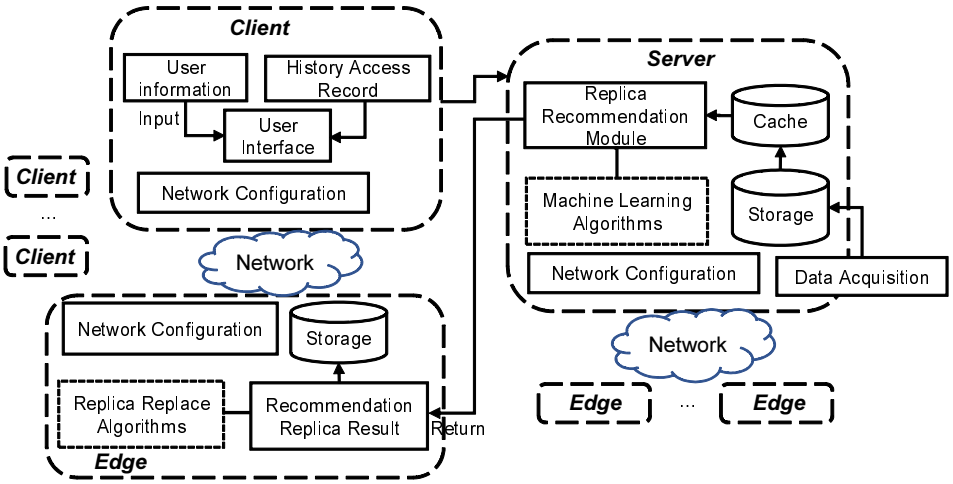

FIGURE 1 The logical view of RPME

Our RPME aims to provide a common framework for placing media replicas into the edge node solution. The logical view of the RPME is shown in Figure 1. The whole system consists of three types of components, namely, the Client, the Server and the Edge node (the device in the middleware that is between the client and the server, such as a switch and a router). The client can access data from the edge nodes and servers, but the replicas obtained by the machine learning algorithms in servers can only be recommended to edge nodes.

The client records the user information and user access history information. The server records all communication information about the edge node and the client in the network configuration. The server can get the data and train the access history information of all users within the communication range of an edge node using machine learning algorithms, such as decision trees and support vector machines (SVM). It can get access to popular content and recommend it to edge nodes. Because the storage capacity of edge nodes is limited, the edge node places the recommended replica according to the proposed optimized replica placement algorithm. At the client and edge nodes, the network configuration stores communication information, according to which the client can connect to and communicate with the edge nodes within the scope of service. Therefore, the client accesses the required data with less delay, which ensures a better user experience throughout the system.

3.2 | Work Flow

The overall architecture of the system is as described above, and the system has four parts. In the first step, according to the demands of users, the client will send a data request to the connected edge node, which can broadcast this request to the whole system. The nearest server will give a clear answer regarding which nodes cached the requested resource to the node through the information in the server. In the next step, this EdgeNode will compare the resource location information that is received, select the nearest node, and retrieve the replica from the node to send back to the user. The above two steps can be clearly shown in Figure 2 (section 1) and Figure 2 (section 2).

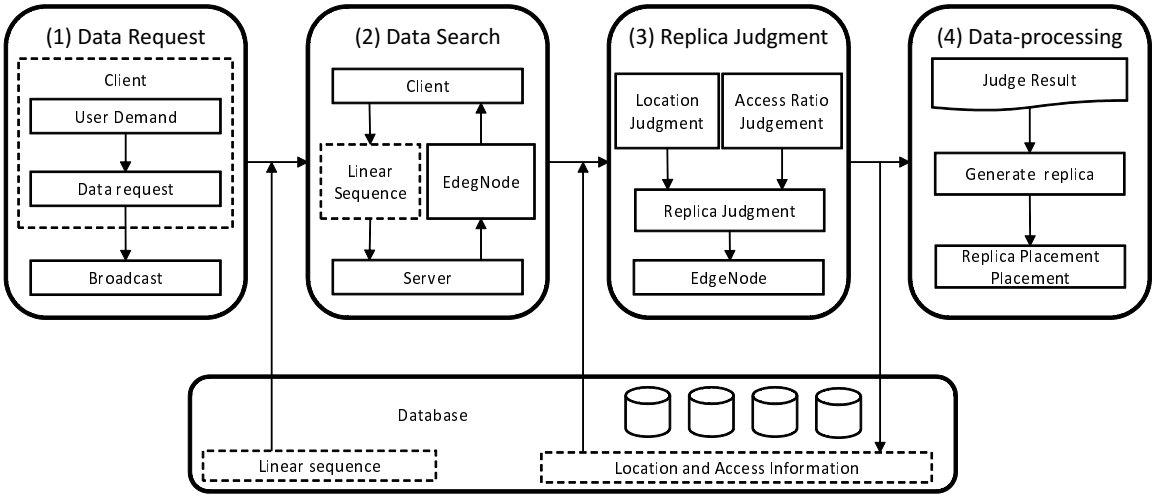

FIGURE 2 System framework flow of RPME

The replica judgement is the third step, which is shown in Figure 2 (section 3). The EdgeNode, which is connected to the requesting client, will determine the location and access the information from the server. According to the results of the two judgements from the information, it will decide whether to place the replica here. In the last section of Figure 2, the edge node will send the calculation results to the server, and it will perform replica placement based on the result. In the whole process, there is a real-time update database in the system. It provides the linear sequence in the second step, gives the replica location and access ratio in the third step and updates the information of the replica in the last step. It is an important part of the whole system.

In the system, when the user requests media data to the server, the user context information and the user-item rating matrix are acquired and transmitted to the server. We first cluster the user on the basis of the user context information and update the user-item rating matrix, and then generate the replica recommendation sequence for the server. When the server pushes the replica recommendation sequence to the edge node, some hot data are placed in the edge node by the proposed the replica placement algorithm. The edge node is able to provide media replicas for multiple clients.

3.3 | Problem Description

For a group of media item the user is requesting $\{I_1, I_2, I_3, ..., I_n\}$, there are the server and the edgeNode to process and response. We hope that the edgeNode can response all requests of the user to improve the user experience. However, because of the limitation of storage capacity of the edgeNode, it is idealized to deploy all the requested data on the edgeNode.

The media replica placement (MRP) problem is to meet the user's personalized demand and gain a better user satisfaction by selecting the most appropriate replica to be placed the edgeNode. On the premise of limited storage capacity on the edge server, the demand (D) for the edgeNode is maximum. The MRP problem is formalized as:

$$D = max \sum_{i=1}^{m} d_i \tag{1}$$

subjected to:

(1) $d_i \geq d_{limination}$;

(2) $\sum_{i=1}^{n} |d_i| \leq |M|$,

$d_{limination}$ is the minimum requirement of the media data, if d_i is lower than $d_{limination}$, the edgeNode will not place the replica of the media data. $|d_i|$ is the size of the media data, and $|M|$ is the storage capacity of the edgeNode.

In fact, we consider the the MRP problem is very difficult to gain the optimum solution in polynomial time. We discuss a feasible heuristic greedy algorithm to solve the MRP problem in this paper.

4 | SOLUTIONS TO THE MRP PROBLEM

In MRP problems, because the historical evaluation of replicas is an important basis for the selection of replicas, users' interests can be predicted according to users' historical scores. Therefore, the recommendation algorithm is introduced into the MRP problem. The recommendation results can satisfy the users' personalized needs and thus have high accuracy. In this section, we first collect user information and analyse some features of the media content accessed by users. On this basis, a recommendation algorithm is proposed to recommend replicas to be selected. In addition, we design a greedy heuristic algorithm to solve MRP problems.

4.1 | Preliminaries

The current situation surrounding users affects the users' behaviours. We first describe the user context abstractly, and the users are divided into several different user groups through clustering analysis. Thus, we could address the problem about cold start-up for the new user.

In the RPME model, user information and Resource information are defined as follows.

Definition 1. User information (UI), including three attributes such as age, sex and occupation, can be represented as follows: UI=(a, s, o), where

(1) a denotes the different age groups, including younger than a certain number of years old and above a certain number of years old;

(2) denotes the different sexes, including man and woman.; and

(3) o denotes the different occupations, including teachers, doctors, and students.

Definition 2. Resource information (RI) can be represented as: RI=(c, r). where:

(1) c denotes the category of resources.

(2) r denotes the user rating of resources.

In RPME model, the user's rating of the resource is obtained by the user's direct explicit rating. Therefore, we can conclude whether the user is interested in the resource. However, user interest resource categories can not be obtained directly. Therefore, in addition to the definition of projects and users, we derive the category of user interest resources based on the type of resources and the user's rating of resources.

Definition 3. User behavior set: Given Γ=$\{K_1,K_2,\cdots,K_m\}$, a user interesting resource categories is a set satisfying: $K_i = \{(u_{identity}, r_{identity})\}$,where:

(1) the m represents the number of resource categories.

(2) u_identity and r_identity seperately represent the identity of the user (or the client) and the resource.

4.2 | Personalized Replica Recommendation Algorithm

To satisfy the user's personalized needs, we propose two recommendation algorithms to choose the appropriate copy to the edge node.

4.2.1 | Decision Tree based Replica Recommendation Algorithm

In this section, we present a decision tree based replica recommendation algorithm to recommend more accurate replica to the edge node.

The basic idea of decision tree is to calculate the information gain of each eigenvalue in the training set according to the eigenvalue vector of the data in the training set, and then select the data with the greatest information gain as the decision point. The other data are divided into two parts

by the decision point. Then the information gain of the two parts of data is calculated separately. Big data divides the original data into four parts. In this way, all data are processed recursively. After processing, the data structure is similar to a binary tree structure, which is called a binary decision tree.

The decision tree is constructed as follows. Each user attribute in the data set is a node in the tree. We divide data sets according to the method of obtaining maximum information gain. After constructing the decision tree, in order to reduce the problem of over matching, we prune the constructed graph decision tree.

The pseudo code of decision tree based replica recommendation algorithm is presented as Algorithm 1. First, the algorithm normalize the user information. Then, the user data is divided into training set and test set. Eventually, The decision tree based replica recommendation algorithm is used to ïňĄnd the common interesting category of each node.

Algorithm 1 Decision Tree based Replica Recommendation Algorithm

Input:N=$n_1, n_2, \cdots, n_p$ // The edge nodes
U=$u_1, u_2, \cdots, u_q$ // The users
I=$i_1, i_2, \cdots, i_q$ // The userinfor of each user
K=$k_1, k_2, \cdots, k_m$ // The category of each movie
Output:cicn // The common interesting category of each node
cicn $\leftarrow \emptyset$;
K(u)$\leftarrow \mathbf{0}$;
normalization of I;
for each n_i $(1 \leq i \leq p)$ **do**
  trs $\leftarrow \emptyset$; tes $\leftarrow \emptyset$;
  split u_j into trs and tes;
  dtf = tree.DecisionTreeClassifier();
  for u_j $(1 \leq j \leq q) \in$ trs **do**
    dtf.fit(I,kind);
  end for
  for u_j $(1 \leq j \leq q) \in$ tes **do**
    predict_kind=dtf. predict(user_info);
    favorite_category=argmax(predict_kind);
    cicn.append(favorite_category);
  end for
end for
return cicn;

Decision tree has low computational complexity, easy to use and high efficiency. Decision tree can process data with irrelevant characteristics and easily construct rules that are easy to understand. However, the decision tree-based replica recommendation algorithm can select the resources that users would like to recommend to edge nodes, it has some shortcomings, such as the difficulty in dealing with missing data, over-fitting and ignoring the correlation between attributes in the data set. In the next section, we further propose a clustering-based replica recommendation algorithm to recommend more accurate replicas to the edge nodes.

4.2.2 | Cluster-based Replica Recommendation Algorithm

Recommending the hot data to the edge node is the key to effectively reducing user access latency. We present a collaborative filtering algorithm based on user context clustering to produce a replica recommendation to meet the user's personalized requirements.

We use the k-means clustering algorithm to assign users of similar contexts to the same class. In this paper, suppose I={i_1,i_2,i_3, ..., i_k } represents a collection of all types, and each user only belongs to one type in the set C. We adopt cosine similarity to calculate the similarity between the user information. sim(μ, ν) is given below [34].

$$sim(\mu, \nu) = cos(\vec{\mu}, \vec{\nu}) = \frac{\vec{\mu} \times \vec{\nu}}{\| \vec{\mu} \| \times \| \vec{\nu} \|} \tag{2}$$

To effectively solve the problem about data sparse of user rating , this paper adopts slope one algorithm [19] to forecast the score filling matrix before collaborative filtering.

We fistly calculate the deviation of the items' rating which is the mean value of the items' rating differential.

$$Dev(i,j) = \frac{\sum\limits_{\mu \in \theta(i) \cap \theta(j)} (r_{\mu_i} - r_{\mu_j})}{\mid \theta(i) \cap \theta(j) \mid} \tag{3}$$

r_{μ_i} is the rating of the item i that the user μ gives, and r_{μ_j} is the rating of the item j that the user μ gives. $\theta(\mathrm{i})$ is the user that overrated on the item i, and $\theta(\mathrm{j})$ is the user that overrated on the item j. $\mid \theta(\mathrm{i}) \cap \theta(\mathrm{j}) \mid$ is the number of the user that overrated on the items i and j.

Then, we predict the score of the item unrated, according to the formula (3) and the user's historical score.

$$P_{\mu_j} = \frac{\sum\limits_{\mu \in \theta(\mu)} \mid \theta(i) \cap \theta(j) \mid (r_{\mu_i} - Dev(i,j))}{\sum\limits_{\mu \in \theta(\mu)} \mid \theta(i) \cap \theta(j) \mid} \tag{4}$$

$\theta(\mu)$ is the item that is overrated by the user μ.

This paper first uses the k-means clustering algorithm to assign users with similar interests to the same class. Then, it predicts the rating of the unscored item according to the slope one algorithm and fills it in using the user-item rating matrix. Eventually, it calculates the similarity of the items using a collaborative filtering algorithm based on the item, updates the rating of the item, and generates the replica recommendation of the item. Details of the algorithm are described as follows, and its pseudo code is given in Algorithm 2.

Algorithm 2 Cluster-based Replica Recommendation Algorithm

```
Input:N=n_1, n_2, ··· , n_j // The edge nodes
      U=u_1, u_2, ··· , u_k // The users
      I=i_1, i_2, ··· , i_k // The user context of each user
      M=m_1, m_2, ··· , m_k // The user context of each user
Output:cicn // The common interesting category of each node
cicn ← ∅;
for each n_i (1 ≤ i ≤ j) do
    kind(u)← 0
    interest ← 0
    for each u_i (1 ≤ i ≤ k) do
        for each c_i (1 ≤ i ≤ t) do
            if c_i==kind( u_i) then
                interest(u_i,c_i)=1;
            end if
        end for
    end for
    feature=(gender, age, occupation,interest)
    normalization of feature
    clf=KMeans(n_clusters)
    clf.fit(feature)
    centers=clf.cluster_centers
    centers_category=category(3:end)
    centers_favorite_category=argmax(centers_category)
    cicn.append(centers_favorite_category)
end for
return cicn;
```

First, using the historical access records of each user on the edge node, you can determine each user's interest resource category and the number of users under each category.

Secondly, feature vectors are obtained according to user information, and feature vectors are normalized. Using the K-Means algorithm to cluster users, we can get the category of common interest resources on each edge node.

Third, we sort the recommended category of interest resources on the edge node and recommend Top-N according to the storage capacity of the edge node.

4.3 | Optimized Replica Placement Algorithm

In addition to the first phase of recommended users 'common interest resource replicas, another important issue is the placement of these recommended replicas. Effective placement algorithms can achieve better load balancing while effectively reducing replica access time. In order to improve the performance of the entire network system.

4.3.1 | Block-level Replica Storage

For each edge node's storage, based on the characteristics of the edge nodes, we propose a block-level replica storage scheme to increase the advantages of the data resource search and to save storage space. Each replica will be split into many blocks. Each replica block will be represented as $CB = tag, record, data$. Tag is the key for the database in the server, and the RPME uses this parameter to search data and access data quickly. Record is the basic information of the replica block for the RPME, including the Last-Time (the latest access time), the Access-Times (the access times), the Block-Index (the index of the block of data), the Next-Block (the location of the next block of data), the Freq (the access frequency in the same area) and the size. The replica is the block context of the replica resource.

The core of the block-level replica is to divide the replica into small blocks. The RPME marks each block of a replica in sequence using the Block-Index and records the location information of the next block as the Next-Block in the block, which allows us to find the full replica through the first block. The storage between blocks is similar to a single linked list of weak links. Therefore, when the replica is large enough and the storage space of one edge node is not to be loaded, the RPME will store the block of one replica in a different edge node. Based on these parameters of data, the RPME can accurately store each replica.

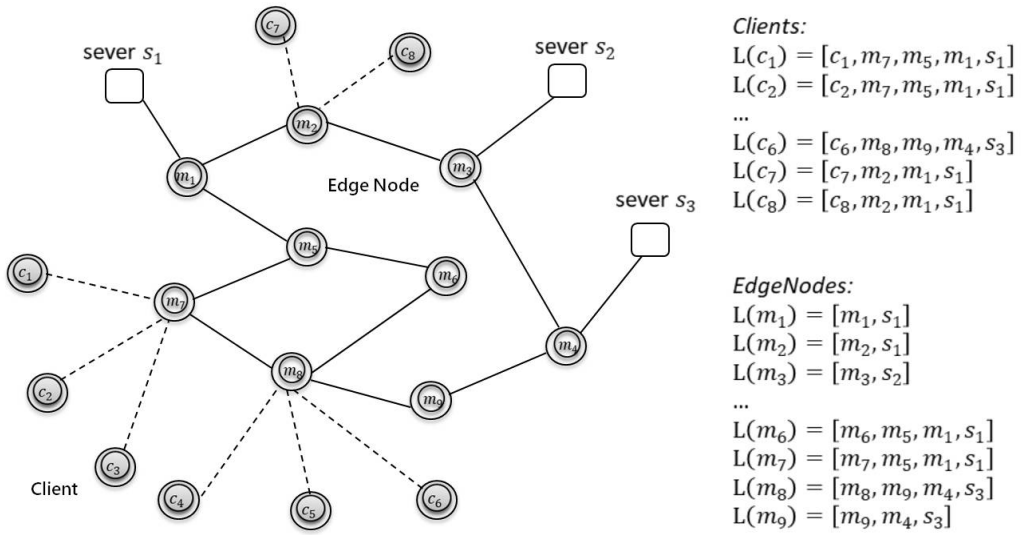

FIGURE 3 Linear sequence of Graph Nodes

Since the replica resources are scattered across the edge node, it is difficult to use the traditional replica search algorithm. To search the locations of replica resources quickly, the RPME has a special index database (Index-DB) in the server. Index-DB records the EdgeNodes where each cache resource is located. In the Index-DB, the RPME does not record the full data replica, and only the location information of the first replica block is recorded. When the EdgeNode starts broadcasting for resources, the broadcast information will be transferred to the server from the nearest to the most distant. According to the Index-DB in the server, we can know whether the requested resource or the replica of this resource exists in the area in which this server is responsible. Until we find the first-block information of the resource, we can feed this information back to the client who sent the request.

In the above mentioned system, we assume that it has three parts: Clients, EdgeNodes and Servers. We can denote them as $C = \{c_1, c_2, ..., c_n\}$, $M = \{m_1, m_2, ..., m_p\}$ and $S = \{s_1, s_2, ..., s_q\}$. According to these nodes, we can define the graphical structure of the network as G and G=(V, E),

where V is all of the vertexes (the nodes in the system) in the communication graph G and E is the edges between the vertexes in G. Each E in the communication graph G is denoted as E=E(w), where w is communication costs. In addition, each V in the communication graph G is denoted as V=V(sc,pc), in which sc is the shared cache size and pc is the private cache size. Therefore, V has different features in the different types of system nodes.

To quickly and accurately find the corresponding nodes, we devise a solution to the problem. We defined a valid data fetch sequence as $L = \{l_1, l_2, \ldots, l_n\}$, in which l_n is a computing component on the path from itself to the server. Every node in the system has its own linear sequence. We can get a communication graph as shown in Figure 3. The linear sequences all start from themselves and end at the nearest server. If there are two servers that have the nearest distance to a component, it will save the first calculated linear sequence to the database. For example, the EdgeNode m_2 has the same distance to server s_1 and server s_2, and it is recorded as the linear sequence $[m_2, s_1]$.

4.3.2 | Optimized Replica Placement Algorithm

This section focuses on media copy placement for edge nodes. For the limit of the storage capacity of the edge node, we assume that each file has only one copy in edgenode. An optimized media copy placement algorithm is proposed to give priority to media copies of interest in order to optimize the matching between the placed copy and the user's needs, so as to maximize the requirements of the edge nodes, meet the user's personality requirements, and improve the user experience.

For edgenode, we consider placing copies while also paying attention to load balancing of edge nodes. In order to improve the resource utilization of edge nodes, a variety of improved greedy algorithms are proposed in algorithm 3. This method considers the real-time load of the edge node and selects the smallest load peer to place the copy. When the copy resource is placed on the edge node, the edge node closest to the request is first selected. Based on the above idea of replication location, our goal is to find a set of nodes to place copies of users 'common interest resources on the premise of limited storage capacity and service capacity to minimize delays in resource access requests and ensure that each user can obtain services. From the above analysis, the replica placement problem can be solved in two steps.

Algorithm 3 Optimized Replica Placement Algorithm

```
Initial:   user_requset, node_number;
Output:  demand,load,delay;
The user of the edge node obeys the Poisson distribution;
for i in range(len(user_requset) do
    for j in range(node_number) do
        if i in node [j] then
            calculate user[i] request node_movie_type[j];
            find node [k] with min_load;
            if user_requset [i] is not found in all node then
                find user_requset [i] from the sever;
            end if
            calculate delay;
        end if
    end for
end for
return demand,load,delay;
```

First, the user arrives at the edge node and obeys the Poisson distribution. Based on user attributes and user history access records, we use the recommendation algorithm proposed in the previous section to recommend the user's common interest resource categories for each edge node. For new resources, we need to determine whether it is the resource category recommended by the edge node. If so, we recommend it to the corresponding edge node.

Secondly, after we determine the resource copy that needs to be placed, we will select a suitable edge node to place the resource copy to ensure the user QoS. Therefore, we propose an optimized greedy algorithm to solve the problem model described in section 3.3 to place resource copies so that the load of each node is balanced.

The key to solving the replica placement problem is to solve the second step: finding a marginal node with storage capacity and service capacity constraints to place resource copies to achieve load balancing for each edge node. Since the replica placement problem is NP-hard, we propose a

greedy algorithm as follows:(1) Find the node that recommends this resource type. (2) Find the node that satisfies the constraint, and(3) prioritize the node with the smallest load to place a replica. Algorithm 3 lists the optimized greedy algorithms.

In the light of the above idea of replica placement, our goal is to find a set of nodes to place replicas of resources under the premise of limited storage capacity and service capacity, so as to minimize the latency of resource access requests to ensure that each user can be served. Therefore, an improved greedy algorithm is proposed.

5 | PERFORMANCE EVALUATION

5.1 | Experiment Settings

We first discuss the impact of media resource replica selection on the replica placement in edge computing. To ensure an effective QoS, we attempt to reduce the access latency by placing the resource replicas on the most suitable edge nodes and mobile clients. In the paper, we ignore the effect of the transmission ratio on the experimental results. We simulated a small edge computing network, which contains two servers, 10 edge nodes and 50 terminals connected to the edge nodes. The regularity of the user's arrival at each edge node obeys the Poisson distribution. This paper adopts the MovieLens data set whose characteristics include Users, Movies, Ratings and Sparsity levels. In addition, users' information can be acquired from the data set. In addition, as the number of nodes increases, the relative distance between nodes and servers becomes farther.

We have employed a simulator to perform a set of experiments to estimate the performance of the presented algorithms through Python. The accuracies of the two recommended algorithms in selecting copies are compared and analysed. We compare the proposed media replica placement algorithm with other algorithms with respect to the replica demand rate, the average response delay and the loading balance. Since the user arrives at the edge node and obeys the Poisson distribution, we take the average of 10 experiments.

5.2 | Recommendation Accuracy

The experiment evaluates the decision tree algorithm and clustering algorithm when selecting the replica by comparing the recommended accuracy. In the RPME model, the recommendation accuracy of resource replica on each edge node is measured by the expectation of the resource copy type requested by all users of the node. Therefore, the accuracy of the recommendation can be defined as follows:

$$accuracy = \frac{1}{n}\sum_{i=1}^{n} E\,(i) \tag{5}$$

, where n is the number of the experiment and E(i) represents the expectation of the resource replica categories requested of all users in each edge node with the range $0 \leq \mathrm{E}\,(\mathrm{i}) \leq 1$.

In the RPME model, the number of categories of recommended resource replicas per node greatly affects the recommendation accuracy. If all resource replica categories are recommended on the edge node, the accuracy is highest. However, this is idealized. Because the storage capacity of the edge node is limited. To obtain the appropriate value for the recommended category, Figure 4 shows the accuracy in different situations.

From Figure 4, we can get two main conclusions:(1) With the increase in the number of recommended resource replicas categories, the accuracy of the recommendation has also improved. (2)The accuracy of clustering algorithm is always higher than decision tree.

The second experiment shows the accuracy of the clustering algorithm and the decision tree algorithm for each edge node when the recommended resource replica category is the same. In the experiment, we randomly select 10 edge nodes and use the above two algorithms to perform 10 experiments respectively, and then calculate the average accuracy. Figure 5 shows that the clustering algorithm has a better recommendation effect than the decision tree algorithm in the RPME model.

5.3 | EdgeNode Load Balancing

In consideration of the replica placement problem, this paper emphasizes the load balancing of edge nodes. For the edge node in the replica location set, we will the select the lower load node to place. In edge computing, the main influencing factors of node load are disk I/O load (L_1) and disk space load (L_2). Its formula is as follow:

$$Load = w_1 * L_1 + w_2 * L_2 \tag{6}$$

Where w_1 and w_2 are the weight values of disk I/O load and disk space load.

We initialize storage space of each edge node and randomly assigned 40 data collections, assuming that each data was the same size. To achieve load balancing of edge nodes, the greedy algorithm is improved. The larger the node load indicates the more busy the disk is and the more space

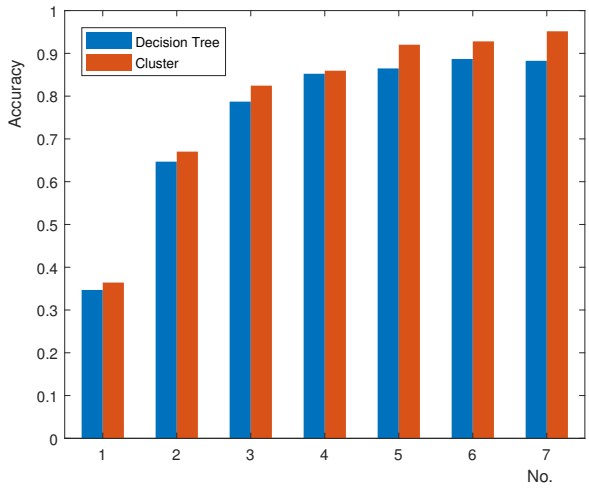

FIGURE 4 Recommendation Accuracy for resource replica categories

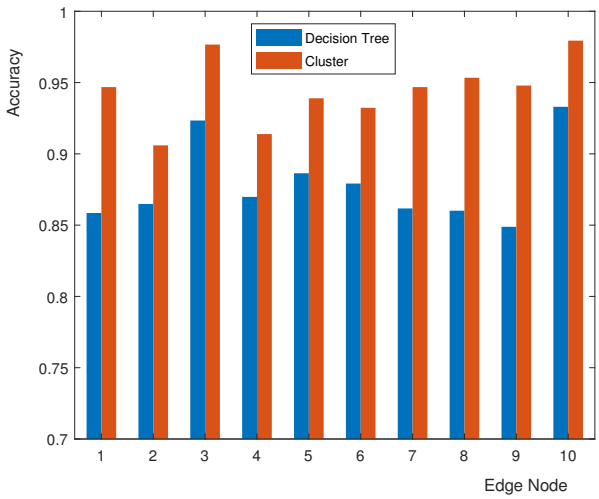

FIGURE 5 Recommendation Accuracy for edge Nodes

is used. Therefore, considering the node load, it can avoid the blocking of the replica block write operation due to the busy disk, which leads to the decrease of the system performance. On the other hand, when balancing the replica distribution as far as possible, it can reduce the pressure of the subsequent copy adjustment operation and save the system resources.

In Figure 6, we compare the load of each edge node with the load balancing for the three algorithms. The experimental results show that the load based on clustering algorithm is the largest. It means that the user's request is most responded to on the edge node. On the contrary, the load of the random algorithm is the lowest, which means that the user's request can not be responded on the edge node and needs to be requested to the server. It will increase the server's load and the edge nodes will not be fully utilized. In other words, the clustering based recommendation algorithm is superior to the other two algorithms.

Furthermore, we compared the load on the edge node in the case of a load-balanced replica placement and a non-load-balanced replica placement. As shown in Figure 7, the load of the edge node is larger in load balancing than in non-load balancing.

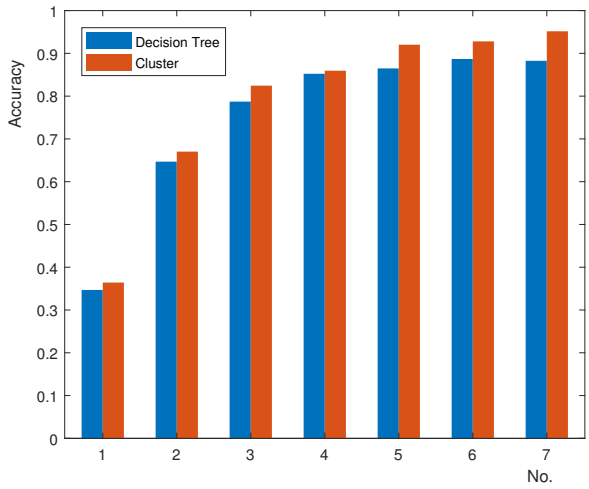

FIGURE 6 Load Balancing for edge Nodes

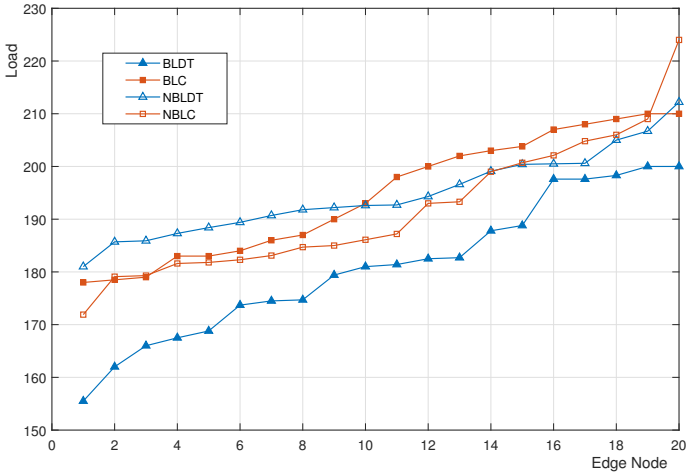

FIGURE 7 Load Balancing and Non-Load Balancing for Edge Nodes

5.4 | Replica Demand Rate

Definition 4. The replica demand rate (D) is the ratio of the number of media replicas that the requesting user just requested from its local edge node with respect to the total number of user requests.

The replica demand rate is represented as follows:

$$D = \frac{N_{edgenode}}{N_{request}} \tag{7}$$

where $N_{edgenode}$ is the number of request responses at the edgeNode, and $N_{request}$ is the total number of requests of all users at the edge node.

As shown in Figure 8, the replica demand rate of the three different algorithms are relatively stable for each edge node when load balancing. The replica demand rate of the cluster algorithm exceeds that of the other two algorithms. The algorithm places replicas on the basis of the recommended replica sequence and the user context, which greatly improves the replica demand rate in comparison with the other algorithms. In the end, the tendency of the replica demand rate becomes relatively steady due to the limitations of the edgeNodeâĂŹs storage capacity and the number of requested replicas.

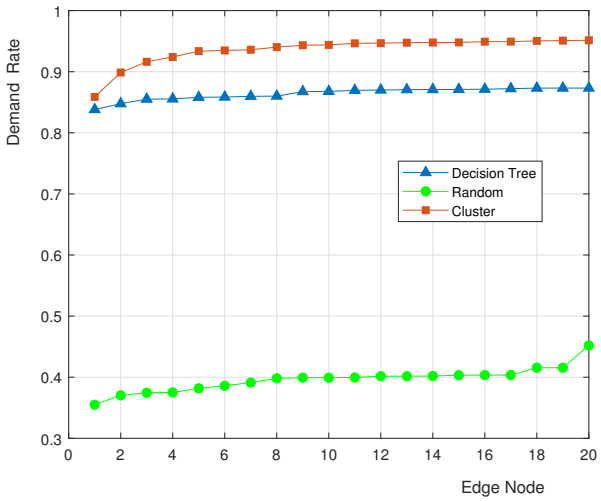

FIGURE 8 Replica demand rate for edge Nodes

5.5 | Average response delay

Definition 5. The average response delay (T_{delay}) is the ratio of the sum of the respective response times for the requested responses at the edgeNode and the server with respect to the total response time of all users.

The average response delay time is represented as:

$$T_{delay} = \frac{\alpha \cdot N_{edgenode} + \beta \cdot N_{server}}{N_{request}} \tag{8}$$

N_{server} is the number of the request responsed at the server. α and β are the delay time on the edgeNode and the server respectively.

For the edge computing, we can calculate the average access time of all users in each edge node to access all the replicas in the database. The average response delay shows the average access efficiency of the total user requests. Thus, we adopt equation (8) to calculate the average response time. Originally, all the data resources are stored on the server side, and the replicas are delivered to each edge node from each server containing the data. The optimized replica placement strategy can reduce the access latency by placing replicas on the edge nodes. Before optimization, we first measured the average access latency of all edge node data. We initialize the storage information on each edge node.

In Figure 9, we compare the delays of the three algorithms on each edge node in the case of load balancing and non-load balancing. The experimental results show that the delay of the three algorithms in load balancing is lower than that of non-load balancing. Moreover, when the load is balanced, the delay of each edge node is roughly the same and relatively stable for three algorithms.

6 | CONCLUSIONS AND FUTURE WORK

In this paper, an optimization mechanism is proposed to solve the problem of media replica placement in edge computing. We first introduce the media replica placement problem in edge computing and highlight its characteristics. Second, we elaborately formulate an edge computing model and the media replica placement problem. Third, in view of the load balancing capability of the edgeNode, the recommendation algorithm using the recommended replica of user information is analysed. Based on the storage and service capabilities of the client, a cooperative replica placement mechanism is proposed to place the resource replicas. Furthermore, the greedy algorithm of the mechanism is implemented.

We have performed many experiments to evaluate the performance of the proposed scheme. The simulation results show that the strategy can reduce the access delay of the client, the load balancing of the server and the replica availability. In future work, we will also consider the network's fault tolerance after replication and updating to ensure the quality of customer service, improve the stability of the network, and reduce network operating costs. The experimental results show that the algorithm has significant replica demand performance.

In future work, we will also focus on some improvements to the proposed algorithm, including experimenting using real network computing environments and improving the performance of the replica requirements.

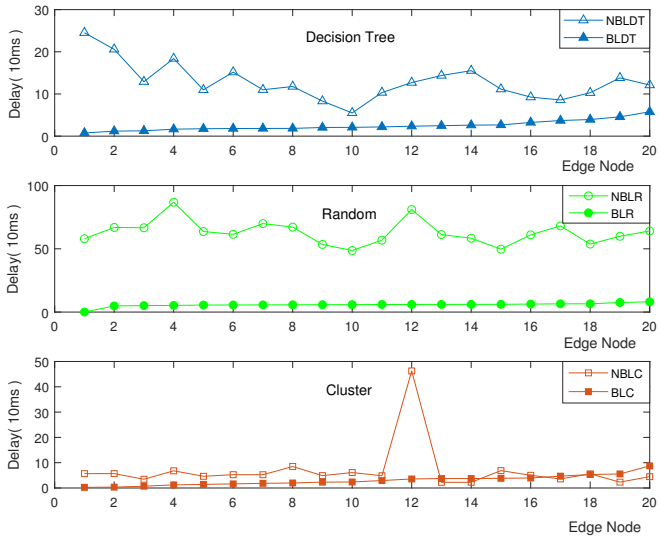

FIGURE 9 Average response delay with different request numbers

ACKNOWLEDGMENTS

This work is supported by the Hunan Science and Technology Plan (2012RS4054), Natural Science Foundation of China (61672535, 61472005), Key Laboratory of Intelligent Perception and Systems for High-Dimensional Information of Ministry of Education Innovation Fund (JYB201502), Key Laboratory of Information Processing and Intelligent Control of Fujian Innovation Fund (MJUKF201735), Natural Science Foundation of Hunan Province (2018JJ3203), Graduate Student Innovation Project of Hunan (CX2016B049), Research Foundation of Education Bureau of Hunan Province (17C0679), Hunan University of Science and Engineering Research Project(17XKY071) and the construct program of applied characteristic discipline in Hunan University of Science and Engineering. The authors declare that they have no conflict of interests.

1 Wang X, Yang L T, Liu H, et al. A big data-as-a-service framework: State-of-the-art and perspectives[J]. IEEE Transactions on Big Data, 2018, 4(3): 325-340.

2 Big Data and Computational Intelligence in Networking[M]. CRC Press, 2017.

3 Cisco. Cisco visual networking index: Global mobile data traffic forecast update 2015-2020 white paper.

4 Vidyarthi, D. P., Sarker, B. K., Tripathi, A. K., & Yang, L. T. (2009). Scheduling in Distributed Computing Systems: Analysis, Design and Models. Springer US.

5 Cheng X, Wu Y, Min G, et al. Network Function Virtualization in Dynamic Networks: A Stochastic Perspective[J]. IEEE Journal on Selected Areas in Communications, 2018.

6 Sheu S T, Huang C H. Mixed P2P-CDN system for media streaming in mobile environment.[C]// International Wireless Communications and Mobile Computing Conference, Iwcmc 2011, Istanbul, Turkey, 4-8 July. 2011:657-660.

7 Wang H T, Song L H. Applications and Challenges of Mobile P2P Systems in Ad Hoc Network[J]. Advanced Materials Research, 2011, 171-172:575-579.

8 Khalaji F K, Analoui M. Replica Placement Algorithms in hybrid CDN-P2P architectures[C]// International Symposium on Telecommunications. 2012:771-775.

9 Tang Y, Guo K, Tian B. A block-level caching optimization method for mobile transparent computing[J]. Peer-to-Peer Networking and Applications. 2018; 11(4): 711-722.

10 Yang, L. T., & Guo, M. (2005). High-performance computing: paradigm and infrastructure (Vol. 44). John Wiley & Sons.

11 Naik, V. K., Liu, C., Yang, L., & Wagner, J. (2005). Online resource matching for heterogeneous grid environments. IEEE International Symposium on CLUSTER Computing and the Grid (Vol.2, pp.607-614 Vol. 2). IEEE.

12 Wang Q, Dai H N, Wang H, et al. Data-Driven QoE Analysis on Video Streaming in Mobile Networks[C]//Ubiquitous Computing and Communications (ISPA/IUCC), 2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on. IEEE, 2017: 1115-1121.

13 Zhu, C., Wang, H., Liu, X., Shu, L., Yang, L. T., & Leung, V. C. M. (2016). A novel sensory data processing framework to integrate sensor networks with mobile cloud. IEEE Systems Journal, 10(3), 1125-1136.

14 Guo K, Liang Z, Shi R, et al. Transparent Learning: An Incremental Machine Learning Framework Based on Transparent Computing[J]. IEEE Network, 2018, 32(1):146-151.

15 Zhao Z, Min G, Gao W, et al. Deploying Edge Computing Nodes for Large-scale IoT: A Diversity Aware Approach[J]. IEEE Internet of Things Journal, 2018.

16 Lili Qiu and V. N. Padmanabhan, "On the Placement of Web Server Replicas", in Proceedings of INFOCOM'01, vol. 3, pp. 1587-1596, April 2001.

17 M. H. Al-Shayeji, S. Rajesh, M. Alsarraf, and R. Alsuwaid, "A Comparative Study on Replica Placement Algorithms for Content Delivery Networks", In 2nd International Conference on Advances in Computing Control and Telecommunication Technologies (ACT), Dec. 2010, pp. 140-142.

18 Jing Sun, Suixiang Gao, Wenguo Yang, Zhipeng Jiang, "Heuristic Replica Placement Algorithms in Content Distribution Networks", in Journal of Networks, Vol 6, No 3 (2011), 416-423, Mar 2011

19 F. L. Presti and C. Petrioli, "Distributed Dynamic Replica Placement and Request Redirection in Content Delivery Networks", 15th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS'07), pp. 366-373, Oct. 2007.

20 Rodrigues, M., Moreira, A., Neves, M., AzevÃłdo, E., Sadok, D., & Callado, A., et al. (2013). Optimizing cross traffic with an adaptive CDN replica placement strategy. Simulation Symposium.

21 Zhiyong Xu and Laxmi Bhuyan, "Qos-Aware Object Replica Placement in CDNs", in Proceedings of IEEE GLOBECOM'05, vol. 2, pp. 862-866, Dec. 2005.

22 Dowdy L W, Foster D V. Comparative Models of the File Assignment Problem.[J]. Acm Computing Surveys, 1982, 14(2):287-313.

23 Fu F G, University N A, Nanjing. An Overlay Topology Based Proactive Replication in Unstructured P2P Systems[J]. Journal of Software, 2007, 18(9):2226-2234.

24 Lin M, Yang L T. Hybrid genetic algorithms for scheduling partially ordered tasks in a multi-processor environment[C]//Real-Time Computing Systems and Applications, 1999. RTCSA'99. Sixth International Conference on. IEEE, 1999: 382-387.

25 Zhou J Y, Song A B, Luo J Z. Evolutionary Game Theoretical Resource Deployment Model for P2P Networks[J]. Journal of Software, 2013, 24(3):526-539.

26 M. Garmehi, M. Analoui. An Economical Mechanism for Multicasting of Content among Servers of Hybrid CDN-PEP Networks[C].6th International Conference on Internet Technology and Secured Transactions, Abu Dhabi, 2011, 566-571

27 Khalaji F K, Analoui M. Hybrid CDN-P2P architecture: Replica con tent Placement Algorithms[C]// Information and Knowledge Technology. 2013:7-12.

28 Hoa D, Silverton T, Fourmaux O, et al. A novel Hybrid CDN-P2P mechanism For effective real-time media streaming[C]// Asia-Pacific Signal and Information Processing Association, 2014 Annual Summit and Conference (APSIPA). IEEE, 2014:1 - 5.

29 Ma M, Wang Z, Su K, et al. Understanding Content Placement Strategies in Smartrouter-based Peer CDN for Video Streaming[J]. 2016.

30 Almashor M, Khalil I, Tari Z, et al. Enhancing Availability in Content Delivery Networks for Mobile Platforms[J]. IEEE Transactions on Parallel & Distributed Systems, 2015, 26(8):2247-2257.

31 Ayyasamy S, Sivanandam S N. A Cluster Based Replication Architecture for Load Balancing in Peer-to-Peer Content Distribution[J]. International Journal of Computer Networks & Communications, 2010, 2(5).

32 Gaber S M, Sumari P. Predictive and content-aware load balancing algorithm for peer-service area based IPTV networks[J]. Multimedia Tools & Applications, 2014, 70(3):1987-2010.

33 Xiong R Q, Luo J Z, Song A B, et al. QoS preference-aware replica selection strategy in cloud computing[J]. Journal on Communications, 2011, 32(7):93-102.

34 Hai-Ling X U, Xiao W U, Xiao-Dong L I, et al. Comparison Study of Internet Recommendation System: Comparison Study of Internet Recommendation System[J]. Journal of Software, 2009, 20(2):350-362.

How to cite this article: Williams K., B. Hoskins, R. Lee, G. Masato, and T. Woollings (2016), A regime analysis of Atlantic winter jet variability applied to evaluate HadGEM3-GC2, *Q.J.R. Meteorol. Soc.*, *2017;00:1–6*.