



Article

A Replica-Selection Algorithm Based on Transmission Completion Time Estimation in ICN

Zhiyuan Wang ^{1,2} , Hong Ni ^{1,2} and Rui Han ^{1,2,*}

- ¹ National Network New Media Engineering Research Center, Institute of Acoustics, Chinese Academy of Sciences, No. 21 North Fourth Ring Road, Haidian District, Beijing 100190, China
- ² School of Electronic, Electrical and Communication Engineering, University of Chinese Academy of Sciences, No. 19(A) Yuquan Road, Shijingshan District, Beijing 100049, China
- * Correspondence: hanr@dsp.ac.cn

Abstract: As the Internet communication model changes from host-centric to content-centric, information-centric networking (ICN) as a new network architecture has received increasing attention. There are often multiple replicas of content in ICN, and how to reasonably utilize the characteristics of multiple replicas to further improve user experience is an important issue. In this paper, we propose a replica-selection algorithm, called the transmission completion time estimation (TCTE) algorithm. TCTE maintains the state of replica nodes in the domain with passive measurements in a limited domain of an enhanced name resolution system (ENRS), then estimates the transmission completion time of different replica nodes and selects the smallest one. When no replica is found in the ENRS domain, the nearest-replica algorithm will be used, so TCTE will not increase the traffic in the core network. Experiments show that TCTE not only effectively improves the user's download rate and edge node throughput, reduces download rate fluctuations, reduces user download delay, and improves fairness, but also has universal applicability.

Keywords: information-centric networking; replica selection; name resolution



Citation: Wang, Z.; Ni, H.; Han, R. A Replica-Selection Algorithm Based on Transmission Completion Time Estimation in ICN. *Future Internet* **2023**, *15*, 120. <https://doi.org/10.3390/fi15040120>

Academic Editors: José Carlos Lopez-Ardao, Miguel Rodríguez Pérez, Sergio Herrería Alonso and Gyu Myoung Lee

Received: 6 February 2023

Revised: 13 March 2023

Accepted: 23 March 2023

Published: 25 March 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

As the number of users increases, the huge demand for higher data rates and bandwidth puts the current Internet under increasing pressure. The communication model of the Internet is changing from a host-centric to a content-centric paradigm [1]. The user's motivation is to obtain the requested content, not to establish a connection with the service provider's device. In the current end-to-end communication model, users need to know the IP of the service provider to establish a connection to transmit content. When a large number of users obtain the same content, all users need to communicate with the server provider's device. Duplicate transmission of content causes a lot of wasted bandwidth. Many ICN architectures have been proposed, such as CCN [2], NDN [3], Mobilityfirst [4], NetInf [5], DONA [6], and SEANet [7]. In recent years, the feasibility of combining ICN with software-defined network (SDN) [8], intent-based networking (IBN) [9], and other technologies has also attracted new attention to ICN.

ICN separates content from its producer's location [10]. Users can complete content acquisition from any node in the network that caches that replica based on the content's name. The caching of replica nodes in the network puts the content closer to the user. Therefore, the traffic in the core network can be effectively alleviated. In ICN, the content is a named data chunk (NDC) with a globally unique name. As the routers in the ICN network can cache the content, there are multiple replicas of content in the network. When a user requests content, how to find and select a suitable replica node to improve the user download rate and network throughput is an important issue, which has been studied by many researchers [11,12]. According to the approach of name resolution, ICN can be divided into two types: the approach of name-based routing (NBR), in which the process

of name routing is the process of replica selection, and the approach of standalone name resolution (SNR). NDN belongs to the former. It forwards the request to the content producer. The replica node can only be selected when it is on the forwarding path, and the replica node outside the path cannot be used. Although many works have been studied, the NBR-based approach is more complex for the process of discovering replica nodes. These approaches either require a cluster head for decision making, or forwarding based on ICN router local information [11]. SNR-based approaches such as [4] can easily obtain a list of network addresses (NA) of replica nodes in the name resolution system (NRS), and then select a replica node in the list. However, the existing replica-selection algorithm does not consider the expected transmission time of NDC. What users need most is to obtain a high and stable transmission rate, so as to have a low transmission completion time.

In this paper, we proposed the novel replica-selection algorithm based on transmission completion time estimation (TCTE). When the replica node is in the ENRS [13] domain, TCTE selects the replica node with the shortest transmission completion time; when the replica node is outside the domain, TCTE selects the nearest replica node. Therefore, TCTE mainly affects the traffic distribution within the ENRS domain without increasing the traffic in the core network.

The main contributions of this paper are as follows:

- To solve the problem of replica node selection in ICN network, we propose the replica-selection algorithm based on the transmission completion time estimation. The transmission completion time of NDC is related to the size of NDC, the expected transmission rate, and the round-trip time (RTT). Therefore, we designed a method to obtain the expected transmission rate, RTT, and NDC size.
- To solve the problem that the replica nodes may not be selected again due to bad performance in a short time, we design an “activation” mechanism.
- We conducted experiments on the proposed algorithm. Experiments show that TCTE not only effectively improves the user’s download rate and edge node throughput, reduces download rate fluctuations, reduces user download delay, and improves fairness, but also has universal applicability.

The remainder of this paper is organized as follows. Section 2 discusses the existing literature on the topic. Section 3 depicts the NDC transmission process and the name resolution service. Section 4 describes our proposed algorithm. Section 5 evaluates the effectiveness of our proposed algorithm. Section 6 concludes our work.

2. Related Works

In ICN networks, content can be cached in different locations, so each piece of content may have multiple replicas, and it is an important issue how to use multiple replicas of ICN effectively. Table 1 shows the characteristics of the related work.

A name resolution service is used to help clients access content, hosts, or services using names, and there are two main approaches, the name-based routing approach (NBR) and the standalone name resolution approach (SNR) [14]. The ICN architecture has different approaches to name resolution services and different issues when performing replica selection.

In the ICN architecture using the name-based routing approach, such as NDN [3], the ICN router needs to select the outgoing interface to forward the content request according to the information it maintains. The process of content request forwarding is the process of replica selection. The shortest-path routing algorithm forwards content requests to content providers, such as OSPFN [15] and NLSR [16], so only replicas on the routing path can be selected. The algorithm has lower overheads, but users have fewer replicas of the content available to them. Many algorithms have been proposed to efficiently utilize content replicas on non-shortest-routing paths. The cluster-based method [17,18], through the cluster head node centralized caching decision and forwarding decision when receiving a content request, can realize the replica selection in the cluster. iNRR [19] explores the network in a scope-flooding approach using the first content request. In SEARCH-CNG [20],

link bandwidth is defined as the ratio of the link's capacity to the number of flows over this link, and the algorithm makes forwarding decisions based on link bandwidth to avoid congested links. Both iNRR and SEARCH-CNG are flood-based methods that explore in-network replicas, thereby reducing traffic on the default forwarding path. INFORM [21] is a dynamic forwarding mechanism based on the Q-learning algorithm [22]. It discovers content replicas through flooding, and then selects forwarding paths based on indicators such as RTT. Stateful forwarding [23] is also a learning-based forwarding mechanism. The ranking value is the basis for the forwarding decision of the algorithm. It represents the priority of the forwarding path, which depends on various performance indicators, such as RTT and link state, etc. In SCAN [24], the single-hop neighbor nodes exchange the information of their cached content regularly through the Bloom filter [25]. When the content request arrives at the node, if the content exists, it will be forwarded directly to the client; if it does not exist, the content will be searched for whether it is cached in a neighbor. RFW [26] is based on random walks. A layered ICN architecture is proposed in RFW, the publisher is located in the first layer, and the client is located in the bottom layer. When the content request arrives at the layer N , the maximum random walking time is T . When the walking time exceeds T , it will be forwarded to the $N - 1$ layer until a replica of the content is found.

In the ICN architectures that use the standalone name resolution approach, such as Mobilityfirst [4] and NetInf [5], the client first obtains the locator of the content from the name resolution system (NRS), then selects a replica, and uses the content locator for routing to complete the content acquisition [27,28]. Compared with the ICN architecture using NBR, the ICN architecture using SNR can easily obtain the locators of multiple content replicas because of the existence of NRS, and then obtain the content. ERS [12] uses the distance-constrained-based name resolution system [13] to discover nearby replicas, and then calculates a node status value based on the maintained replica node queuing delay, outstanding requests, RTT, and the network distance for replica selection. Mobilityfirst uses a global name resolution system (GNRS) to obtain the locator of the replica, and then sends the content request to the nearest replica node according to the routing table. NetInf obtains the locators of all replica nodes, and selects the replica node according to factors such as network distance and delay.

In a distributed system, the problem of replica node selection also exists. Algorithms can be divided into information-independent algorithms and information-aware algorithms. Information-independent algorithms mainly include fixed strategies, random strategies, and cyclic strategies. The advantage of this type of algorithm is that it does not require perceptual information, so the overhead is low, but with the static replica-selection strategy it is often difficult to achieve the optimal effect [29]. The information-aware algorithm actively or passively measures information such as RTT, bandwidth, and loads of different replica nodes, and selects replicas based on the measurement information. Carter et al. [30] measure the available bandwidth and round-trip time (RTT) of different replica nodes, and estimate the time required for content download to guide the selection of replica nodes. However, since the algorithm needs to actively perform network detection, it will occupy additional network bandwidth. Ping-random [31] selects a set containing the five best-performing replica nodes based on ping times, and the client randomly selects a replica node in the set when making replica selection. The two random choices (2RC) [32] algorithm randomly selects two replica nodes and chooses the less-loaded node to process the request.

The approach of NBR-based ICN architecture is not suitable for NRS-based ICN architecture. MF's nearest-replica approach and ERS are suitable for the ICN architecture of NRS, but they do not consider the NDC download rate and NDC download delay that users are most concerned about. The approaches in traditional networks are often not directly applicable to ICN. This paper proposes a new replica-selection approach suitable for the ICN architecture of NRS. The main idea of TCTE is similar to the literature [30], which is based on the estimated value of the transmission completion time for replica

selection. However, their replica node information maintenance and replica-selection steps are completely different.

Table 1. Characteristics of the related work.

Application Scenarios	Approaches	Characteristics
NBR-based ICN architecture	Shortest-path routing	Only replicas on the path can be selected.
	Cluster-based method	Cluster head node decision.
	iNRR	Select the nearest replica by flooding.
	SEARCH-CNG	Avoid selecting replicas on congested paths.
	INFORM	Use flooding to discover replicas, and complete replica selection based on indicators such as RTT.
	Stateful forwarding	Calculate the ranking value of the link and select the replica on the better link.
	SCAN	The information exchange of the cache node can select the neighbor replica of the node on the path.
	RFW	Applicable to a hierarchical ICN architecture, the lower-level replica is selected by random walk.
NRS-based ICN architecture	MF	Select the closest replica based on a routing table.
	ERS	Based on weighted values of congestion, RTT, and hops.
	NetInf TP	Based on latency and network distance. (No approach details provided by author.)
Traditional network	Reference [30]	Proactively detects the network and selects the replica with the lowest flow completion time.
	Ping-random	Randomly selects one of the five smallest replicas of RTT.
	2RC	Randomly takes two replica nodes and chooses the one with the smaller load.

3. NDC Transmission and NRS Overview

Due to the existence of a large number of IP infrastructures, the evolutionary deployment of ICN networks is a realistic consideration [33]. The proposed replica-selection algorithm is based on the ICN architecture compatible with IP infrastructure. In this section, we introduce the named data chunk (NDC) transmission process and name resolution system (NRS) of the model.

3.1. Overview of the NDC Transmission Process

In ICN, each piece of content is an NDC whose size is reasonable, from hundreds of KBs to a few MBs [34,35]. The ICN router caches the NDC to serve as a replica node. The identifier (ID) and the locator in the ICN are separated, the identifier can identify the NDC and the device in the network, and the locator is the network address (NA). The client sends a content request (REQ) packet to the network, and the replica node responds to the REQ packet and replies with a content data (DATA) packet. Both the REQ packet and the DATA packet will carry a source ID and a destination ID. In the REQ packet, the source ID identifies the client device, and the destination ID identifies the NDC. In the DATA packet, the source ID identifies the NDC, and the destination ID identifies the client device. An NDC transmission process can be identified according to the device ID and the NDC ID.

Due to the consideration of flow and congestion control, the replica node cannot send all the data of NDC at once, otherwise it will cause traffic bursts in the network, and even cause the network to crash. The REQ packet will carry the segmentation information of the data, indicating the starting offset and length of the requested data in the NDC. Clients can perform receiver-driven congestion control by controlling the size of the requested data per REQ packet and the frequency at which REQ packets are sent. In this paper, we use Copa-ICN [36] as the congestion control algorithm of the client, which has a good performance in terms of algorithm convergence speed and fairness. In addition, packet loss

is hard to avoid during transmission. When data loss occurs, the client will re-request the lost data to ensure reliable transmission of NDC.

When the replica node receives the REQ packet, it will read the NDC data from the cache according to the request data segmentation information in the REQ packet and assemble it into the DATA packet to send. Due to the limitation of the maximum transmission unit (MTU), the replica node may split the data requested by an REQ into several DATA packets and send them. In addition, the ICN router can use the late-binding technology [13], and if the current replica node cannot retrieve the cache of the NDC, the replica node can initiate a query to the NRS and forward its REQ packet to another replica node until the REQ packet reaches the replica node that caches its requested NDC.

Figure 1 is a schematic diagram of the NDC transmission process. Due to the support of the late-binding technology, when the client acquires an NDC, it can directly send an REQ packet to the edge node without performing name resolution. After receiving the first REQ packet, the edge node will request a network address (NA) list of replica nodes from the NRS, and use the replica-selection algorithm to select a replica node RN3 (in Figure 1) to forward the REQ packet. After receiving the REQ, RN3 replies with the DATA packet according to the REQ. The DATA packet will be forwarded to the client. It should be noted that the client can obtain the size of the NDC only after receiving the first DATA packet.

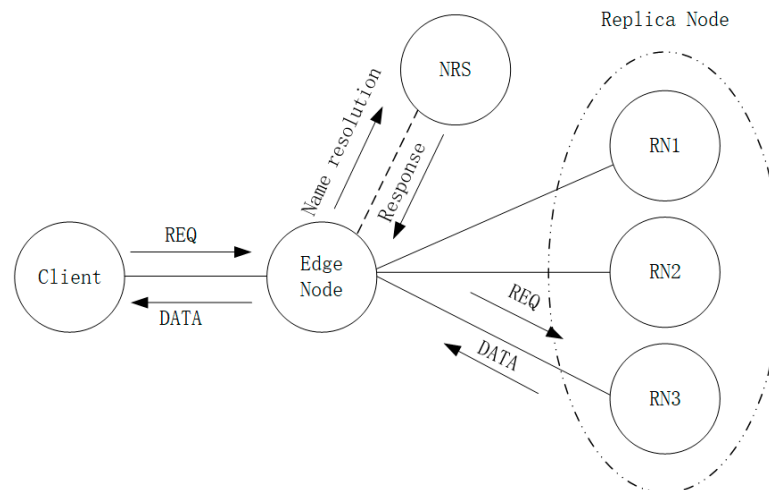


Figure 1. Diagram of NDC transmission process.

3.2. Name Resolution System

Name resolution services are provided by the name resolution system (NRS). NRS types include the enhanced name resolution system (ENRS) [13] and global name resolution (GNR). ENRS is a name resolution service of the SNR approach. It partitions space into hierarchical domains based on distance constraints. No two domains at the same level overlap. ENRS can implement low-latency name resolution services within a limited domain, while inter-domain name resolution services are provided by global name resolution (GNR). The number of layers of ENRS has no effect on our study of replica-selection algorithms, so in this paper, we use a single layer of ENRS to implement our algorithm.

Figure 2 is a schematic diagram of a single-layer ENRS. The ICN router will register its network address (NA) and cached NDC ID in the ENRS and GNR of its domain. When obtaining content, the content-requesting entity will first query ENRS for a node NA list with the content according to the ID of the content. If the query result is empty, it will continue to query GNR. Finally, the replica-selection algorithm is used to select a replica node for data transmission.

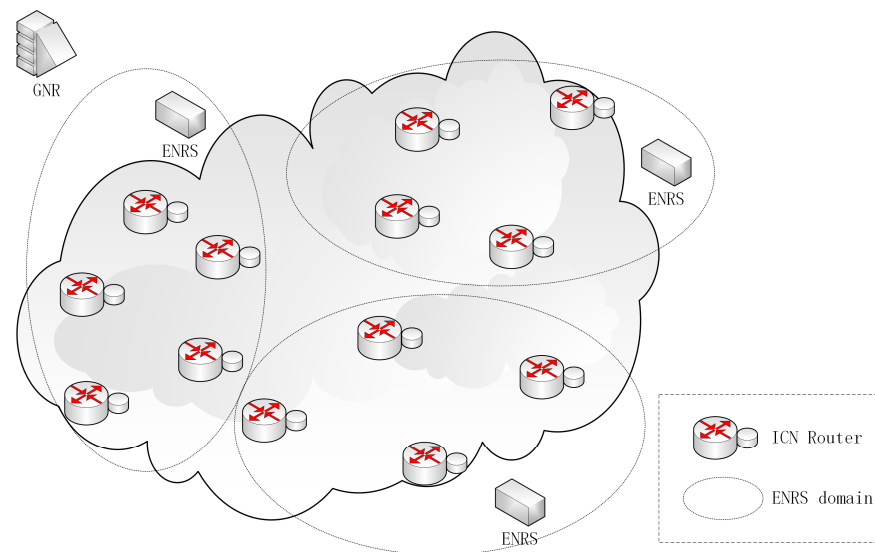


Figure 2. ENRS schematic diagram.

4. Algorithm Description

In this section, we describe the TCTE algorithm in detail and analyze the algorithm overheads.

4.1. Motivation

In an ICN, the replica-selection algorithm will directly affect the user experience. A better replica-selection algorithm can effectively increase the user's download rate and ensure a relatively stable rate, which is conducive to improving the overall throughput of the network.

Figure 3 is a schematic diagram of replica selection. Assuming that the user to R1 node is not the bottleneck, if RN1 and RN2 can be used reasonably, the speed of users' downloading NDC can be improved. In addition, users and replica nodes (RN1, RN2) are all in the same ENRS domain, so it will not increase the traffic in the core network. Considering the replica-selection algorithm based on the estimation of transmission completion time, if the estimated value of transmission completion time of the replica node of RN2 is lower than that of RN1, RN2 will be selected to provide services. After RN2 is selected once, when the replica selection is performed again, because some resources of RN2 are occupied, the estimated value of the transmission completion time may be higher than that of RN1, and RN1 will be selected at this time. By continuously selecting replica nodes with smaller transmission completion time estimates, a reasonable distribution of traffic among different replica nodes can be achieved. Therefore, we will propose a replica-selection algorithm based on transmission completion time estimation, called TCTE. TCTE enables better replica selection based on transmission completion time estimates of replica nodes in the domain.

4.2. Overview of TCTE Algorithm

Compared with the traditional IP network, ICN enables users to obtain content from a closer location through the cache replica in the network, which not only improves the user experience, but also effectively reduces the traffic of the core network, which is conducive to improving the overall performance of the network. Implementing a replica-selection algorithm based on transmission completion time estimation can improve user experience, but the replica node with the smallest estimated transmission completion time may also be a far-away replica node, which will increase traffic in the core network. In addition, it is extremely difficult and expensive to maintain replica node information within the entire network. Therefore, we implement the approach based on transmission completion time estimation within the ENRS [13] domain. On the one hand, a set of replica nodes returned

by ENRS is in the same ENRS domain as the user, without obtaining a replica node that is far away. On the other hand, the overhead of maintaining replica node information in the ENRS domain is controllable. TCTE performs more complex replica-selection operations within the ENRS domain. If no replicas can be found within the ENRS domain, the TCTE will take the replica nodes obtained from the GNR. In this case, we use the nearest-replica selection algorithm as in Mobilityfirst to avoid increasing the traffic in the core network. Performing replica selection requires better awareness of the network, but the information that the client can perceive is limited. Edge routers [37] can aggregate user requests, while, due to the large number of data passing through they can better perform passive measurements on the network. Furthermore, because of the support of late-binding technology, it is a reasonable choice to run the replica-selection algorithm on the edge nodes. Inspired by the work of [12], we maintain the status of the replica nodes in the ENRS domain at the edge nodes.

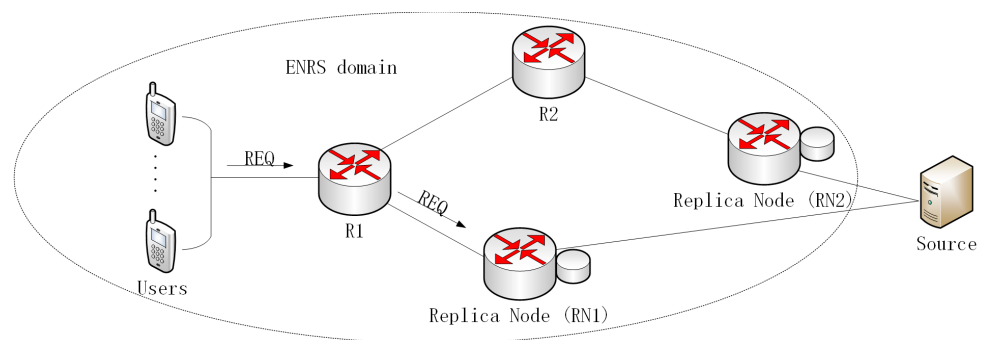


Figure 3. Schematic diagram of replica selection.

When a replica node is selected, we try to estimate the transmission completion time T for the NDC.

$$T = a * RTT + b * \frac{ChunkSize}{Rate}, \quad (1)$$

where RTT is the round-trip time between the edge node and the replica node. For a replica node in the ENRS domain, we set a timeout value of 10 s (or other reasonable values) for its RTT . When the RTT of a replica node expires, five REQ packets forwarded to the replica node will be marked for re-measurement. Take the minimum value among the measured values as the RTT of the replica node. $ChunkSize$ is the size of NDC, and $Rate$ is the expected transmission rate of the replica node. Parameters a and b are adjustable coefficients. $Rate$ is calculated based on passive measurements of edge nodes and will be described in detail in Section 4.3. $ChunkSize$ needs to complete the transmission of at least one DATA packet before it can be obtained. Therefore, before estimating the T value, it is necessary to first select a replica node to obtain a DATA packet. In order to obtain $ChunkSize$ as soon as possible, the first REQ packet of NDC will be forwarded to the replica node with the smallest RTT . After the edge node receives the first DATA packet and obtains the $ChunkSize$ of the NDC, it reselects the replica node with the smallest T value for NDC transmission, and saves the selection result in the replica-selection completion list. When receiving the REQ packet of the transmission process again, the edge node will directly obtain the replica node NA in the replica-selection completion list to forward the REQ packet.

The TCTE algorithm runs on the edge nodes. Algorithm 1 is the operation when TCTE receives the REQ packet. When receiving the REQ packet, TCTE first checks whether there is information about the completion of the replica selection of the NDC transmission process. If it exists, it means that the replica selection has been completed, and it then obtains the selected replica node NA and forwards the REQ. If it does not exist, the replica selection is performed. First, it retrieves the ID of the NDC from the REQ packet, and then

obtains the NA list *NAList* of the replica node from the ENRS according to the ID. If the *NAList* is not empty, it selects the replica node with the smallest RTT to forward the REQ, saves the *NAList*, and marks that replica selection needs to be re-selected. If *NAList* is empty, it continues to obtain the *NAList* of the replica node from GNR. Then, it selects the nearest replica node to forward REQ, and saves the selection result in the replica-selection completed list.

Algorithm 1: Operation of the TCTE algorithm when receiving an REQ packet

Input: REQ
Output: NA

```

1:  NA = GetNAFromReplicaSelectionCompleteList(REQ)
2:  if NA != NULL then
3:    return NA
4:  else
5:    EID = GetEID(REQ)
6:    NAList = GetNameResolutionFromENRS(EID)
7:    if NAList != NULL then
8:      Reselection = true
9:      NA = FindMinRTTNA(NAList)
10:     SaveNAList(NAList, REQ, Reselection)
11:   else
12:     NAList = GetNameResolutionFromGNR(EID)
13:     if NAList != NULL then
14:       NA = FindNearestNA(NAList)
15:     end if
16:     SaveToReplicaSelectionCompleteList(REQ, NA)
17:   end if
18: end if
19: return NA

```

Algorithm 2 is the operation when TCTE receives the DATA packet. When receiving the DATA packet, TCTE will check whether it needs to re-select the replica node. If it is necessary to re-select the replica node, TCTE will take out the previously saved *NAList* and obtain the *ChunkSize* from the DATA packet, then calculate the estimated value of the transmission completion time of the replica node in the *NAList* according to the status information of the replica node, and select the replica node with the smallest value and save it in the replica-selection completed list.

Algorithm 2: Operation of the TCTE algorithm when receiving a DATA packet

Input: DATA
Output: NULL

```

1:  if is Reselection (DATA) then
2:    NAList = GetNAList(DATA)
3:    ChunkSize = GetChunkSize(DATA)
4:    Tmin = MaxVal
5:    for NAtmp in NAList then
6:      Ttmp = EstimatedTransmissionCompletionTime(NAtmp, ChunkSize)
7:      if Tmin > Ttmp then
8:        NA = NAtmp
9:        Tmin = Ttmp
10:     end if
11:   end for
12:   SaveToReplicaSelectionCompleteList(DATA, NA)
13: end if
14: return

```

4.3. Estimated Transmission Rate of Replica Nodes

We perceive the changes in the transmission rate of the replica nodes by passive measurements. For the measurements of the transmission rate, we smooth based on the time interval between two rate measurements, to prevent large jitters in the rate estimate. In

addition, in order to solve the problem that the replica node may not be selected again due to poor performance in a short period of time, we also designed an “activation” mechanism.

We maintain the $RateRN$ value of the replica node, which represents the total rate of DATA packets transmitted between the edge node and the replica node.

Therefore, the transmission rate $Rate$ of the replica node is:

$$Rate = \frac{RateRN}{\max(N_c, 1)}, \quad (2)$$

where N_c is the number of NDC transmission processes that the current edge node uses the replica node.

$RateRN$ consists of two parts:

$$RateRN = RateRN_{real} + RateRN_{extra}, \quad (3)$$

where we update $RateRN_{real}$ based on the measurement of the transmission rate of DATA packets between the replica node and the edge node.

$$RateRN_{real} = RateRN_{real} * (1 - X(t)) + \frac{DataSum}{t} * X(t), \quad (4)$$

where t is the time since the last $RateRN_{real}$ update. $DataSum$ is the total size of DATA packets received within time t . $X(t)$ is a function about t : $X(t) = c * t / (1 + c * t)$ (c is an adjustable constant). $X(t)$ can dynamically adjust the weight of the historical value and the current measurement value according to the time t . When t is larger, the validity of the historical value is lower, so the weight of the current measurement value is larger. Conversely, historical values have a larger weight to smooth changes in the $RateRN_{real}$ value.

When the $Rate$ calculated by the replica node is too small to be selected again, its $RateRN_{real}$ will become 0, because there is no new data transmission. Even if its link condition improves, we cannot perceive it. Therefore, we add $RateRN_{extra}$:

$$RateRN_{extra} = \begin{cases} RateRN_{extra} + t * d, & \text{if } RateRN_{real} = 0 \\ 0, & \text{if } RateRN_{real} \neq 0 \end{cases} \quad (5)$$

where t is the time since the last $RateRN_{extra}$ update, and d is an adjustable constant.

In this way, even if there is no new data transmission, since $RateRN_{extra}$ increases over time, the replica node will be reselected after a period of time, thus being “activated”.

4.4. Overhead Analysis

We will analyze the communication overhead, memory overhead, and computation overhead of running the TCTE algorithm on edge nodes. TCTE uses passive measurement to perceive the link status between the replica node and the edge node, and does not send additional packets into the network, so the TCTE algorithm does not bring additional communication overhead. The memory overhead of TCTE is $O(n)$. The more replica nodes in the ENRS domain, the greater the memory overhead of TCTE. In our implementation, TCTE needs to occupy about 200 bytes of memory for a replica node. However, even maintaining the state of 10,000 replica nodes only requires a few MB of overhead, and within a reasonable ENRS domain, it will not cause a large burden on memory. The number of replica nodes in the domain is n . The maximum value of TCTE computation overhead for completing a replica selection is $O(n)$. When a certain NDC has caches in all replica nodes, the computational overhead of TCTE reaches the maximum. However, most of the time only a few replica nodes have the same replica, so the maximum cost of a replica selection will be far less than $O(n)$. Therefore, the additional overhead of TCTE is acceptable.

5. Performance Evaluation

In order to verify the effectiveness of the TCTE algorithm, in this section, we use the NS-3 simulator for simulation experiments. In previous work, the ICN transport protocol has been implemented in NS-3 [38]. The transmission protocol is receiver-driven and the congestion control algorithm is Copa-ICN, which can complete the data transmission from a replica node in the network based on the ID of the chunk [36]. We implemented TCTE in the edge node (router). In addition, to illustrate the effect of TCTE, we also implemented the intra-domain random selection algorithm (IDRS) with the nearest-replica algorithm (NR) [4,12].

5.1. Experimental Setup

In the experiments, we randomly generate 1000 named data chunks (NDC) between 100 KB and 10 MB, and each chunk has a globally unique ID. These chunks are randomly placed among replica nodes within the ENRS domain. The replica node registers the mapping between the cached chunk ID and its NA in the NRS. There are several users under the edge node to continuously obtain named data chunks. Each time the user acquires a chunk, a chunk will be acquired again after the transmission of the previous chunk is completed. After several tests, a is set to 2 and b is set to 1 in expression (1). The value of c in function $X(t)$ is set to 0.005. The user's request follows the Zipf distribution [37,39]. The probability that the NDC i is requested is $q_i = e * i^{-\alpha}$, $e = 1 / (\sum_{i \in O} i^{-\alpha})$, and the value of α indicates the degree of concentration of the request. In the experiment, the value of α is 1. We set up two different experimental scenarios to evaluate the performance of the TCTE algorithm. The replica nodes and edge nodes in the experiments are both in the same ENRS domain.

Figure 4 is the topology of Experimental Scenario 1. In this scenario, different replica nodes do not share bottleneck links (the link between the user and the edge node is not the bottleneck). It contains fifteen replica nodes and one edge node. Between the edge node and each replica node, between zero and five routing nodes are randomly generated. The link bandwidth between every two nodes on the path from the edge node to the replica node is a random number between 20 Mbps and 100 Mbps, and the link delay is set to 2 ms. A total of 60 users are set under edge nodes, and users continue to send requests within the experimental time. The simulation experiment time is 100 s.

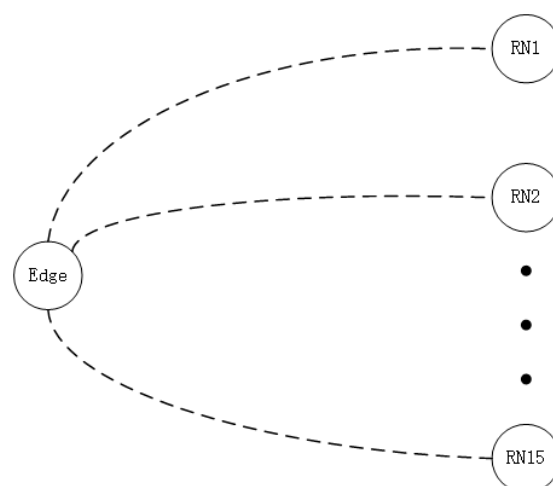


Figure 4. Experimental Scenario 1 topology (Black dots represent replica nodes RN3 to RN14).

Figure 5 is the topology of Experimental Scenario 2. In this scenario, some replica nodes share the bottleneck link, and some replica nodes do not share the bottleneck link. It is a four-layer network topology containing fourteen replica nodes and one edge node. The bandwidth of each link is a random value, as shown in Table 2. The link between the user and the edge node is not the bottleneck. The latency of each link is 2 ms. A total of 60 users

are set under edge nodes, and users continue to send requests within the experimental time. The simulation experiment time is 100 s.

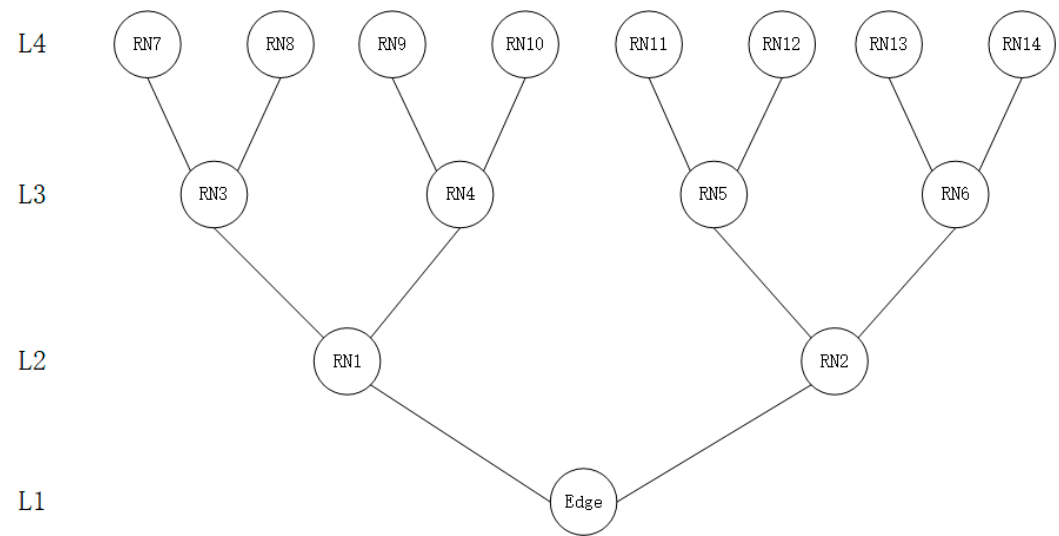


Figure 5. Experimental Scenario 2 topology.

Table 2. Bandwidth settings for Experiment Scenario 2 topology.

Link	Bandwidth (Mbps)
Between L1 and L2	Random values (20 to 100)
Between L2 and L3	Random values (100 to 200)
Between L3 and L4	Random values (200 to 300)

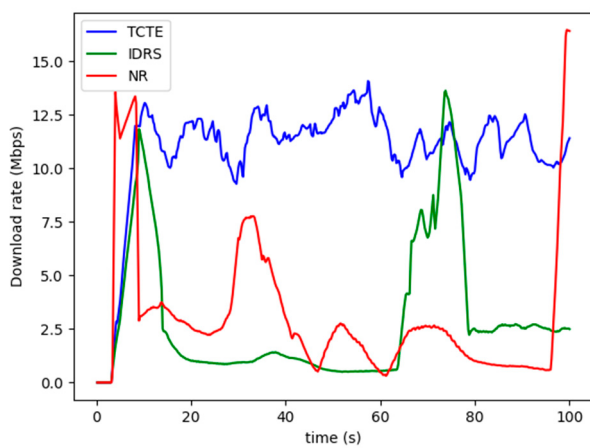
5.2. Performance Comparison

5.2.1. User's Download Rate

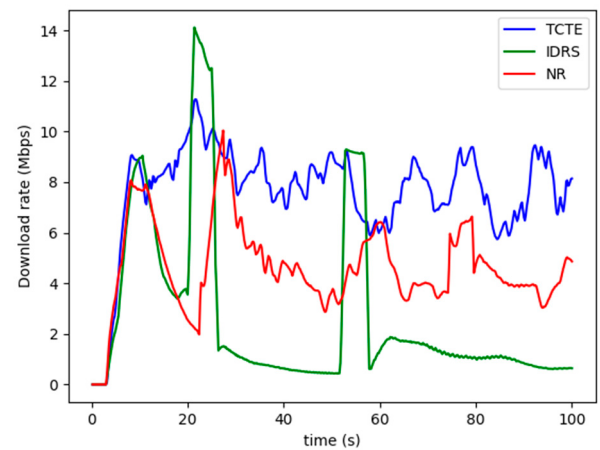
Figure 6a shows the variation in the download rate over time for a user in Experiment Scenario 1. We can see that the TCTE algorithm has the highest user download rate most of the time, while the IDRS and NR algorithms only have the highest user download rate very few times. In addition, the TCTE algorithm has a more stable download rate than the other two algorithms, and the other two algorithms have large fluctuations in the user download rate during the experimental time. Figure 6b shows the variation in a user's download rate over time in Experimental scenario 2. We can see that the TCTE algorithm has the highest download rate most of the time, while IDRS and NR perform poorly. Moreover, the download rate of the TCTE algorithm still has the most stable download rate. Therefore, the TCTE algorithm can enable users to obtain a higher and more stable download rate. In addition, in order to illustrate the reliability of the experimental results, we repeated the experiment 10 times, and took the average download rate of each experiment for analysis. Figure 7 shows the mean and confidence interval (95% confidence level) for the download rate.

5.2.2. Edge Node Throughput

We measured throughput at the edge node. Figure 8a shows the variation in throughput over time in Experimental Scenario 1. TCTE has much higher throughput than the other two algorithms, followed by the IDRS algorithm, and the NR algorithm is the worst. This is because it is difficult for the IDRS and NR algorithms to reasonably distribute traffic to several replica nodes that do not share bottleneck links. The NR algorithm cannot select a farther replica node, even if the farther replica node can obtain a higher transmission rate. IDRS randomly selects replica nodes, which can somehow achieve dynamic distribution of traffic on different paths, so IDRS performs better than NR. However, it is difficult to achieve the best effect by random selection.



(a) Experiment Scenario 1.



(b) Experiment Scenario 2.

Figure 6. User download rate over time.

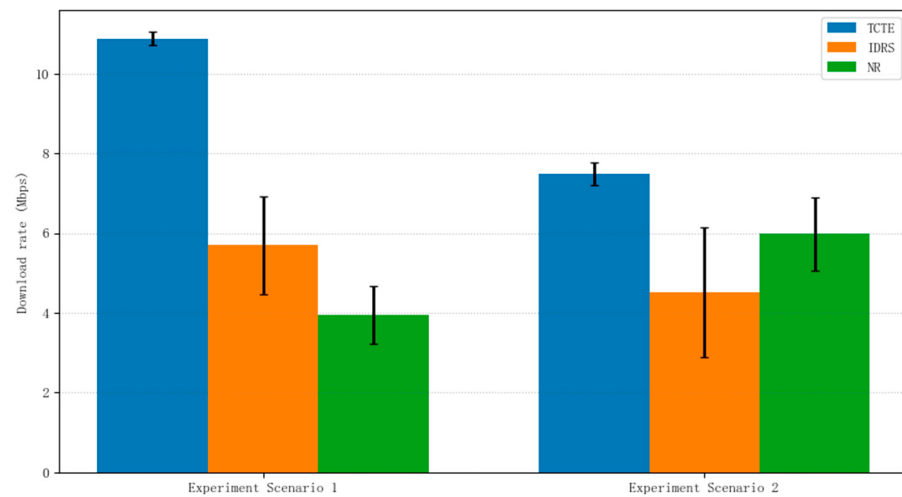
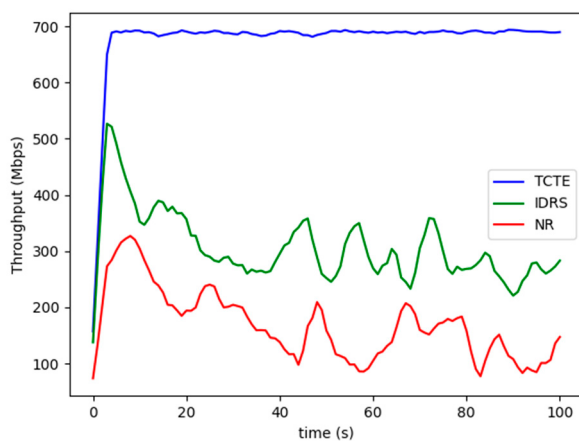
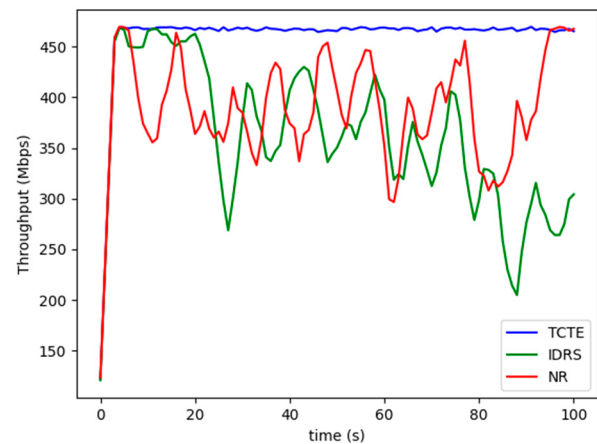


Figure 7. Mean and confidence intervals for download rates.



(a) Experiment Scenario 1.



(b) Experiment Scenario 2.

Figure 8. Throughput of edge nodes.

Figure 8b shows the variation in throughput over time in Experimental Scenario 2. The highest throughput is still the TCTE algorithm, while IDRS and NR perform poorly. The performance of NR becomes better because the nearer replica node in this scenario has

a higher probability of being an intermediate node between the farther replica node and the edge node. The bottleneck bandwidth between the edge nodes and the intermediate nodes is not lower than that between the edge nodes and the farther nodes. In addition, in both scenarios, TCTE has a very stable throughput, while the throughput of the other two algorithms has large fluctuations. Therefore, TCTE not only effectively improves the throughput of edge nodes, but also keeps the throughput stable.

In addition, in order to illustrate the reliability of the experimental results, we repeated the experiment 10 times, and took the average edge node throughput of each experiment for analysis. Figure 9 shows the mean and confidence interval (95% confidence level) for edge node throughput.

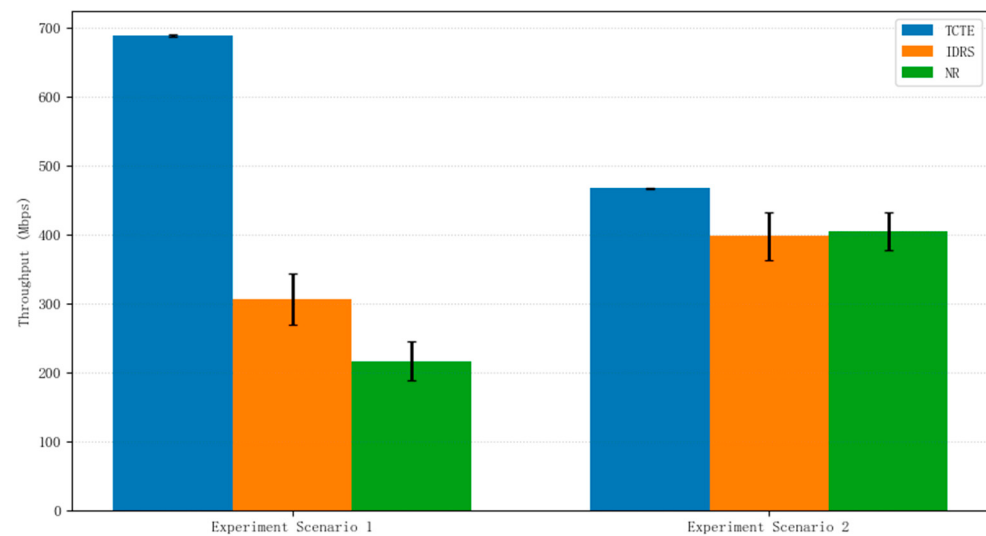


Figure 9. Mean and confidence intervals for edge node throughput.

5.2.3. Fairness

We use Jain's Fairness Index (JFI) to evaluate the fairness of the algorithm. JFI was proposed in [40], and its expression is:

$$F(x) = \frac{(\sum x_i)^2}{n(\sum x_i^2)} \quad (6)$$

where x_i is the transmission rate of user i . n is the number of users. The larger the $F(x)$ value, the better the fairness.

Figure 10 shows the variation in JFI over time in two experimental scenarios. TCTE always has the best fairness performance, while IDRS and NR have poor fairness performance. NR and IDRS perform similarly in Experimental Scenario 1, and NR outperforms IDRS in Experimental Scenario 2. IDRS and NR have high fairness at the beginning of the experiment, and then decline and stabilize at a small value. Unlike IDRS and NR, TCTE has a low JFI for a period of time at the beginning of the experiment, and then stabilizes at a value close to 1. As there were fewer data at the beginning of the experiment, TCTE could not perceive the network condition well.

In addition, in order to illustrate the reliability of the experimental results, we repeated the experiment 10 times, and took the JFI for analysis. Figure 11 shows the mean and confidence interval (95% confidence level) for JFI.

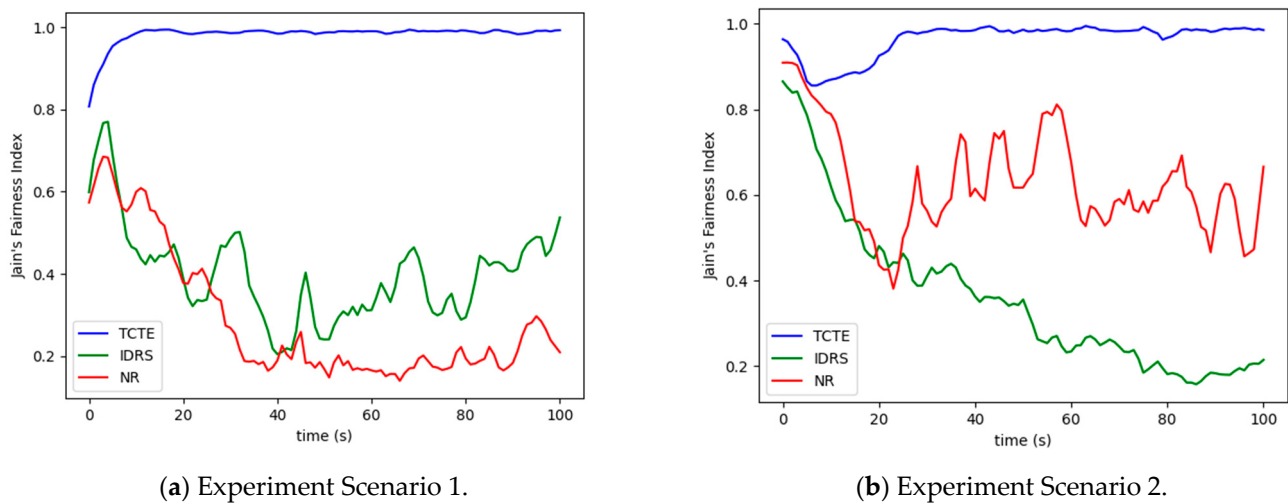


Figure 10. Variation in Jain's Fairness Index over time.

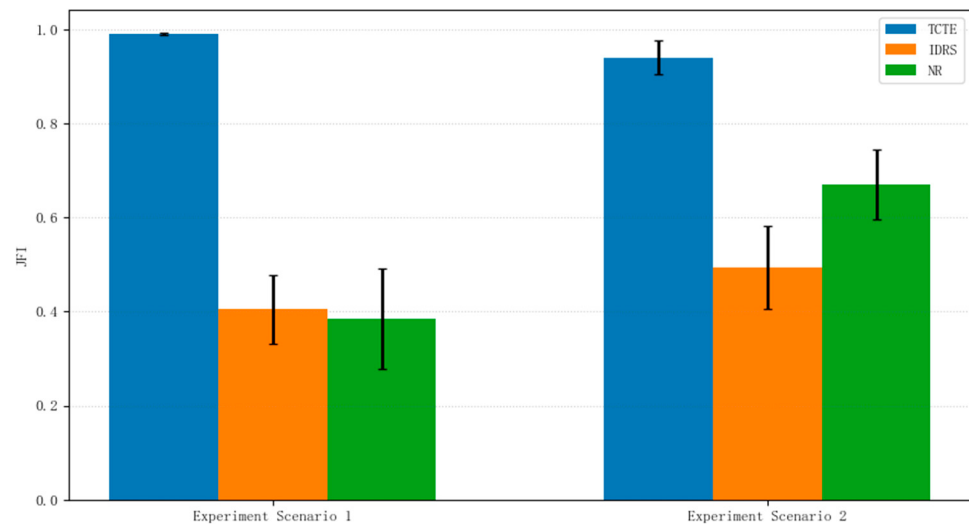


Figure 11. Mean and confidence intervals for JFI.

5.2.4. NDC Download Delay

In order to observe the relationship between NDC download delay and the number of users, we modified the number of users under the edge nodes in Experimental Scenario 1 and Experimental Scenario 2 for experiments. The number of users was 30, 40, 50, 60, 70, 80, 90, and 100. In addition, in order to make the experimental results universal, each experiment was repeated 10 times and the average and standard deviation of the average download delay of 10 experiments were extracted.

Figure 12 is the result of Experimental Scenario 1. The download delay of the TCTE algorithm is the lowest, the IDRS algorithm is second, and the NR algorithm is the worst. With the increase in the number of users, the download delay of the three algorithms continued to increase, and the gap became larger and larger. Figure 12b is a change in the standard deviation. It can be seen that TCTE has the most stable performance, and the performance of the other two algorithms has relatively large fluctuations.

Figure 13 is the result of Experimental Scenario 2. The download delay of the TCTE algorithm is the lowest, and the NR and DRS algorithm download delay is large. With the increase in the number of users, the download delay of the three algorithms continued to increase. Figure 13b is a change in the standard deviation. It can be seen that TCTE has the most stable performance, and the performance of the other two algorithms has relatively large fluctuations. Different from Experimental Scene 2 is that the NR algo-

rithm is better than IDRS, and the gap between NR and TCTE download delay does not significantly expand.

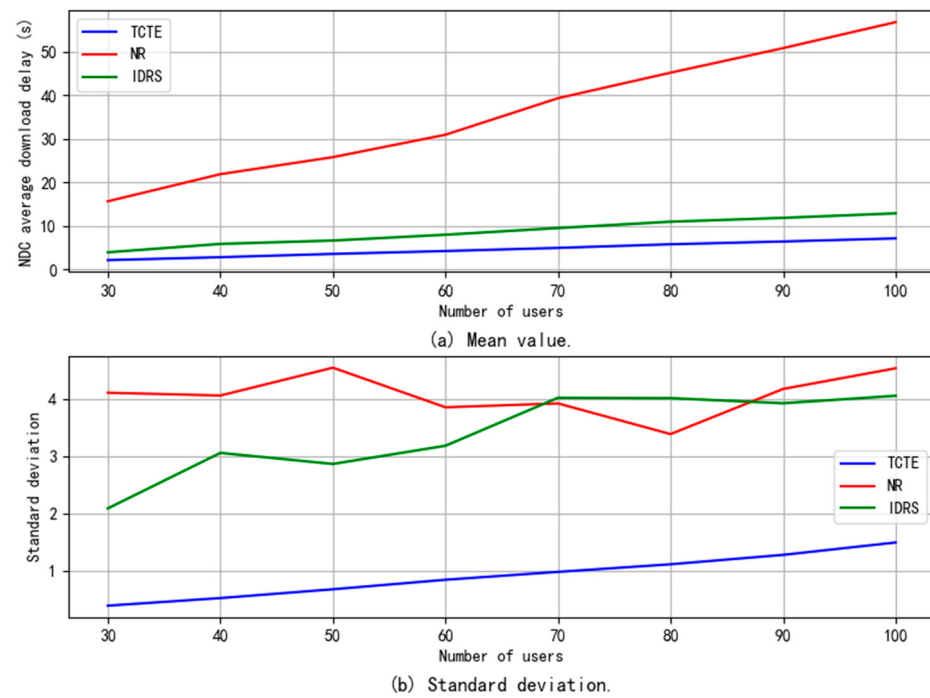


Figure 12. Mean and standard deviation of the NDC average download delay in Experimental Scenario 1.

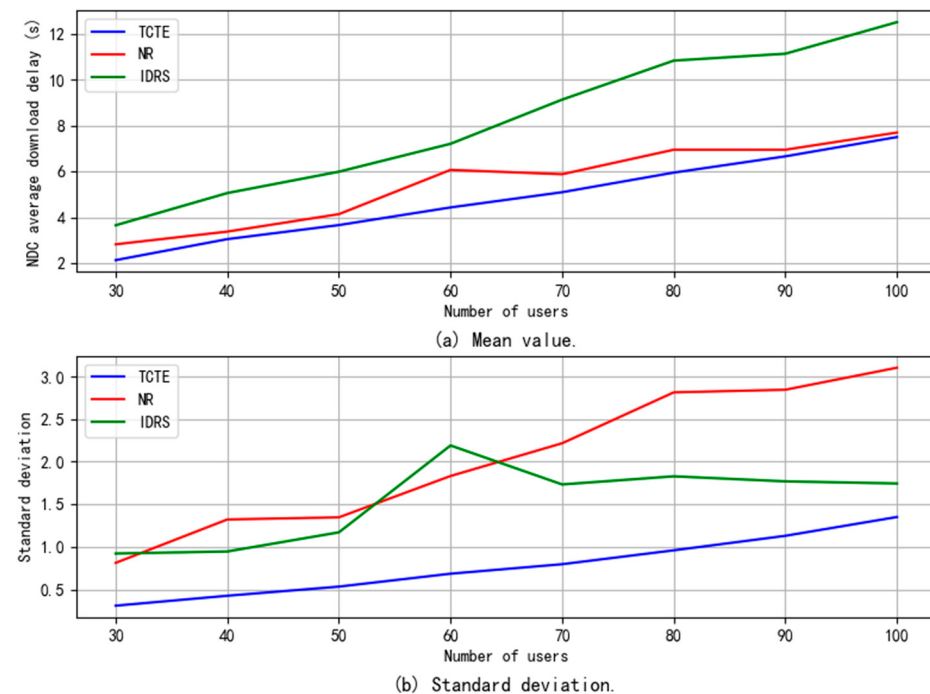


Figure 13. Mean and standard deviation of the NDC average download delay in Experimental Scenario 2.

Based on the results of the two experimental scenarios, TCTE always has the smallest NDC average download delay average and standard deviation, while NR and IDRS always have a large average and standard deviation. This means that NR and IDRS can only perform well in very few experiments, but TCTE has the best performance in most cases.

To illustrate the effectiveness of the experimental results and analysis in Sections 5.2.1–5.2.3, we conducted a t-test on the NDC download delay when the number of users is 60. Table 3 shows the results, and the p -value retains four significant figures. It can be seen that the maximum p -value in experimental scenario 1 is 4.512×10^{-3} , so the experimental results are significantly different at the 99.55% confidence level. In experimental scenario 2, the maximum p -value is 3.952×10^{-2} , so the experimental results are significantly different at the 96.05% confidence level. Therefore, the experimental results are statistically significant and were not obtained by chance.

Table 3. p -values of the t-test for the average NDC download delay with 60 users.

Experimental Scenario	TCTE-IDRS	IDRS-NR	NR-TCTE
Experimental Scenario 1	8.771×10^{-6}	4.512×10^{-3}	4.994×10^{-10}
Experimental Scenario 2	6.637×10^{-4}	2.350×10^{-2}	3.952×10^{-2}

6. Conclusions

In this paper, we propose an innovative approach based on transmission completion time estimation (TCTE) for the replica-selection problem in ICNs to ensure user experience and improve the throughput of the network. TCTE maintains the replica node information in the ENRS domain through passive measurement, and then estimates the transmission completion time and selects the replica node with the smallest transmission completion time. It should be noted that our algorithm only affects the distribution of traffic in the ENRS domain, and will not increase the traffic of the core network. Experiments show that TCTE not only effectively improves the user's download rate and edge node throughput, reduces download rate fluctuations, reduces user download delay, and improves fairness, but also has universal applicability.

Author Contributions: Methodology, Z.W., H.N. and R.H.; software, Z.W.; writing—original draft preparation, Z.W.; writing—review and editing, Z.W., H.N. and R.H.; supervision, R.H.; project administration, R.H.; funding acquisition, H.N. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the Strategic Leadership Project of Chinese Academy of Sciences: SEANET Technology Standardization Research System Development (Project No. XDC02070100).

Data Availability Statement: Not applicable.

Acknowledgments: We would like to express our gratitude to Jinlin Wang for their meaningful support in this work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Nour, B.; Sharif, K.; Li, F.; Yang, S.; Mounsla, H.; Wang, Y. ICN publisher-subscriber models: Challenges and group-based communication. *IEEE Netw.* **2019**, *33*, 156–163. [CrossRef]
2. Jacobson, V.; Smetters, D.K.; Thornton, J.D.; Plass, M.F.; Briggs, N.H.; Braynard, R.L. Networking named content. In Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies, Rome, Italy, 1–4 December 2009; pp. 1–12.
3. NDN. The Named Data Networking Project. Available online: <http://www.named-data.net/> (accessed on 9 January 2023).
4. Raychaudhuri, D.; Nagaraja, K.; Venkataramani, A. Mobilityfirst: A robust and trustworthy mobility-centric architecture for the future internet. *ACM Sigmob. Mob. Comput. Commun. Rev.* **2012**, *16*, 2–13. [CrossRef]
5. Dannewitz, C.; Kutscher, D.; Ohlman, B.; Farrell, S.; Ahlgren, B.; Karl, H. Network of information (netinf)—an information-centric networking architecture. *Comput. Commun.* **2013**, *36*, 721–735. [CrossRef]
6. Koponen, T.; Chawla, M.; Chun, B.-G.; Ermolinskiy, A.; Kim, K.H.; Shenker, S.; Stoica, I. A data-oriented (and beyond) network architecture. In Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, Kyoto, Japan, 27–31 August 2007; pp. 181–192.
7. Wang, J.; Chen, G.; You, J.; Sun, P. SEANet: Architecture and Technologies of an On-site, Elastic, Autonomous Network. *J. Netw. New Media* **2020**, *6*, 1–8.

8. Saadeh, H.; Almobaideen, W.; Sabri, K.E.; Saadeh, M. Hybrid SDN-ICN architecture design for the internet of things. In Proceedings of the 2019 Sixth International Conference on Software Defined Systems (SDS), Rome, Italy, 10–13 June 2019; pp. 96–101.
9. Mehmood, K.; Krlevska, K.; Palma, D.J. Intent-driven autonomous network and service management in future networks: A structured literature review. *arXiv* **2021**, arXiv:2108.04560. [\[CrossRef\]](#)
10. Xylomenos, G.; Ververidis, C.N.; Siris, V.A.; Fotiou, N.; Tsilopoulos, C.; Vasilakos, X.; Katsaros, K.V.; Polyzos, G.C. A survey of information-centric networking research. *IEEE Commun. Surv. Tutor.* **2013**, *16*, 1024–1049. [\[CrossRef\]](#)
11. Ioannou, A.; Weber, S. A Survey of Caching Policies and Forwarding Mechanisms in Information-Centric Networking. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 2847–2886. [\[CrossRef\]](#)
12. Song, Y.; Ni, H.; Zhu, X. An Enhanced Replica Selection Approach Based on Distance Constraint in ICN. *Electronics* **2021**, *10*, 490. [\[CrossRef\]](#)
13. Liao, Y.; Sheng, Y.; Wang, J. A deterministic latency name resolution framework using network partitioning for 5G-ICN integration. *Int. J. Innov. Comput. Inf. Control* **2019**, *15*, 1865–1880.
14. Dong, L.; Wang, G. A Hybrid Approach for Name Resolution and Producer Selection in Information Centric Network. In Proceedings of the 2018 International Conference on Computing, Networking and Communications (ICNC), Maui, HI, USA, 5–8 March 2018; pp. 574–580.
15. Wang, L.; Hoque, A.; Yi, C.; Alyyan, A.; Zhang, B. OSPFN: An OSPF Based Routing Protocol for Named Data Networking. Available online: <https://named-data.net/techreport/TR003-OSPFN.pdf> (accessed on 5 February 2023).
16. Hoque, A.M.; Amin, S.O.; Alyyan, A.; Zhang, B.; Zhang, L.; Wang, L. NLSR: Named-data link state routing protocol. In Proceedings of the 3rd ACM SIGCOMM Workshop on Information-Centric Networking, Kyoto, Japan, 26–28 September 2016; pp. 15–20.
17. Yan, H.; Gao, D.; Su, W.; Foh, C.H.; Zhang, H.; Vasilakos, A.V. Caching Strategy Based on Hierarchical Cluster for Named Data Networking. *IEEE Access* **2017**, *5*, 8433–8443. [\[CrossRef\]](#)
18. Hasan, K.; Jeong, S.H. Efficient Caching for Delivery of Multimedia Information with Low Latency in ICN. In Proceedings of the 2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN), Split, Croatia, 2–5 July 2019; pp. 745–747.
19. Rossini, G.; Rossi, D. Coupling caching and forwarding: Benefits, analysis, and implementation. In Proceedings of the 1st ACM Conference on Information-Centric Networking, Osaka, Japan, 19–21 September 2022; pp. 127–136.
20. Badov, M.; Seetharam, A.; Kurose, J.; Firoiu, V.; Nanda, S. Congestion-aware caching and search in information-centric networks. In Proceedings of the 1st ACM Conference on Information-Centric Networking, Osaka, Japan, 19–21 September 2022; pp. 37–46.
21. Chiochetti, R.; Perino, D.; Carofiglio, G.; Rossi, D.; Rossini, G. Inform: A dynamic interest forwarding mechanism for information centric networking. In Proceedings of the 3rd ACM SIGCOMM Workshop on Information-Centric Networking, Kyoto, Japan, 26–28 September 2016; pp. 9–14.
22. Watkins, C.J.; Daya, P. Technical Note: Q-Learning. *Mach. Learn.* **1992**, *8*, 279–292. [\[CrossRef\]](#)
23. Yi, C.; Afanasyev, A.; Moiseenko, I.; Wang, L.; Zhang, B.; Zhang, L. A case for stateful forwarding plane. *Comput. Commun.* **2013**, *36*, 779–791. [\[CrossRef\]](#)
24. Lee, M.; Cho, K.; Park, K.; Kwon, T.; Choi, Y. SCAN: Scalable content routing for content-aware networking. In Proceedings of the 2011 IEEE International Conference on Communications (ICC), Kyoto, Japan, 5–9 June 2011; pp. 1–5.
25. Bloom, B.H. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM* **1970**, *13*, 422–426. [\[CrossRef\]](#)
26. Domingues, G.d.M.B.; Leão, R.M.M.; Menasché, D.S. Enabling information centric networks through opportunistic search, routing and caching. *arXiv* **2013**, arXiv:1310.8258.
27. Sevilla, S.; Mahadevan, P.; Garcia-Luna-Aceves, J. iDNS: Enabling information centric networking through The DNS. In Proceedings of the 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Toronto, ON, Canada, 27 April–2 May 2014; pp. 476–481.
28. Fuller, V.; Farinacci, D. RFC 6833: Locator/ID Separation Protocol (LISP) Map-Server Interface; RFC, Ed.; ACM Digital Library: New York City, NY, USA, 2013.
29. Bogdanov, K.; Peón-Quirós, M.; Maguire, G.Q., Jr.; Kostić, D. The nearest replica can be farther than you think. In Proceedings of the Sixth ACM Symposium on Cloud Computing, Kohala Coast, HI, USA, 27–29 August 2015; pp. 16–29.
30. Carter, R.L.; Crovella, M.E. Server selection using dynamic path characterization in wide-area networks. In Proceedings of the INFOCOM’97, Kobe, Japan, 7–12 April 1997; pp. 1014–1021.
31. Hanna, K.M.; Natarajan, N.; Levine, B.N. Evaluation of a novel two-step server selection metric. In Proceedings of the Ninth International Conference on Network Protocols ICNP 2001, Riverside, CA, USA, 11–14 November 2001; pp. 290–300.
32. Mitzenmacher, M. The power of two choices in randomized load balancing. *IEEE Trans. Parallel Distrib. Syst.* **2001**, *12*, 1094–1104. [\[CrossRef\]](#)
33. Zeng, L.; Ni, H.; Han, R. An incrementally deployable IP-compatible-information-centric networking hierarchical cache system. *Appl. Sci.* **2020**, *10*, 6228. [\[CrossRef\]](#)
34. Salsano, S.; Detti, A.; Cancellieri, M.; Pomposini, M.; Blefari-Melazzi, N. Transport-layer issues in information centric networks. In Proceedings of the Second Edition of the ICN Workshop on Information-Centric Networking, Helsinki, Finland, 17 August 2012; pp. 19–24.

35. Song, Y.; Ni, H.; Zhu, X. Analytical Modeling of Optimal Chunk Size for Efficient Transmission in Information-Centric Networking. *J. Int. J. Innov. Comput. Inf. Control* **2020**, *16*, 1511–1525.
36. Wang, Z.; Ni, H.; Han, R. Copa-ICN: Improving Copa as a Congestion Control Algorithm in Information-Centric Networking. *Electronics* **2022**, *11*, 1710. [[CrossRef](#)]
37. Saino, L.; Psaras, I.; Pavlou, G. Hash-routing schemes for information centric networking. In Proceedings of the 3rd ACM SIGCOMM Workshop on Information-Centric Networking, Kyoto, Japan, 26–28 September 2016; pp. 27–32.
38. NS-3 Project. Available online: <https://www.nsnam.org> (accessed on 9 January 2023).
39. Breslau, L.; Cao, P.; Fan, L.; Phillips, G.; Shenker, S. Web caching and Zipf-like distributions: Evidence and implications. In *The Future Is Now* (Cat. No. 99CH36320), *Proceedings of the IEEE INFOCOM'99 Conference on Computer Communications, Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies, New York, NY, USA, 21–25 March 1999*; IEEE: Piscataway, NJ, USA, 1999; pp. 126–134.
40. Chiu, D.-M.; Jain, R. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Comput. Netw. ISDN Syst.* **1989**, *17*, 1–14. [[CrossRef](#)]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.