Journal of Information Hiding and Multimedia Signal Processing ©2019 ISSN 2073-4212
Ubiquitous International Volume **10**, Number **2**, March **2019**

A New Node Selection Method for the Ceph Storage System based on Software-defined Network and Fuzzy Multi-attribute Decision-making

Yong Wang
Key Laboratory of Cognitive Radio and Information Processing
Guilin University of Electronic Technology, Guilin China 541004
School of Computer Science and Information Security
Guilin University of Electronic Technology, Guilin China 541004

Yiming Huan
School of Information and Communication
Guilin University of Electronic Technology, Guilin China 541004

Miao Ye*
Key Laboratory of Cognitive Radio and Information Processing
Guilin University of Electronic Technology, Guilin China 541004
College of Information Science and Engineering
Guilin University of Technology, Guilin China 541006

Received June 2018; revised November 2018

Abstract. *In multicopy block storage mode, the Ceph distributed cloud storage system only uses the remaining capacity of nodes as the criterion for selecting storage nodes. However, the network and host conditions are not taken into account in Ceph storage system. When facing fluctuations in the quality of links and increasing loading of storage nodes, the read and write performance of the cluster will be reduced. To optimize the efficiency and quality of storage, we propose a node selection method for a Ceph storage system based on SDN and fuzzy multiple-attribute decision-making method, which can take the network and host condition as multi-factors with remaining capacity to select storage nodes. The designed approach uses software-defined network technology to measure the load status and link qualities of the storage nodes in real time and uses them as decision factors. We formulate our problem with multi-decision factors as a fuzzy multi-attribute decision-making model and solve it to find a group of storage nodes with not only sufficient capacity but also better network conditions and host performance. The test results of the storage node selection method in a real environment demonstrate that the proposed improved node selection scheme effectively reduces the response time for read requests and improves the storage efficiency of the cluster.*
Keywords: Cloud storage; Ceph; Software-defined network; Fuzzy multi-attribute decision-making; Node selection

1. Introduction.

In the era of big data, as the most basic internet service, data storage changing from direct attachment to distributed storage and distributed storage systems has gradually become a research focus for enterprises and research institutes [1]. For example, Google developed the Google File System (GFS) [2] as the dedicated storage system for Hadoop, while Sage Weil designed a distributed file system known as Ceph [3]. The Ceph distributed storage system has now become widely used. Ceph implements its node selection method using the Controlled Replication Under Scalable Hashing (CRUSH) algorithm [4]. However, CRUSH works on a pseudo-random distribution, only considering the remaining capacity of nodes at a particular moment. When the cluster is in service, link quality metrics such as link bandwidth, time delay, and the loading condition of storage nodes may change at any time. If the node selection method is only based on remaining capacity, it would lead to non-optimal storage node selections and a decline in the performance of the overall cluster. Conversely, if the storage system considers the whole network state and load conditions in its node storage selection method, the read/write performance of the whole storage system can be optimized.

To utilize the network state information, it is necessary to measure the network information accurately and efficiently. Traditional methods of network state measurement require extensive configuration and incur significant instrumentation costs [4] [5]. Software Defined Network (SDN, Software Defined Network) [6] [7] is a new programmable network model, in which the control layer and data layer are separated. By enhancing the control of the whole system using the control layer, the efficiency of data transmission, control, and management in large-scale cluster storage is significantly improved. Hence, a reduction in hardware configuration and measurement costs can be achieved if the SDN-based network measurement technology is applied to the Ceph distributed storage system.

This work focuses on designing a Ceph storage cluster scheme based on SDN [6]. The remaining capacity of nodes is considered in the selection method defined by the CRUSH algorithm but our scheme will also use SDN to assess the quality of links, as well as the loading conditions. Then, a fuzzy multi-attribute decision-making mathematical model with multiple parameters [8] is formulated and solved to obtain the OSD affinity of storage nodes. The OSD affinity will then be considered in the node selection process.

The contributions of this paper are as follows:

(1) We design and implement a Ceph framework in combination with SDN to realize a dynamic scheme for obtaining link information and mainframe performance parameters among nodes to inform node selection and increase storage efficiency.

(2) We apply fuzzy multi-attribute decision-making, using the network information parameters and mainframe information from SDN as the input to the Ceph mapping algorithm. Then, we solve the fuzzy multi-attribute abstract matrix to ensure optimal storage node selection.

(3) The information acquisition module based on SDN as well as the node selection module deployed in Ceph is designed and implemented to optimize the node selection method of Ceph.

This paper is organized as follows: Section 2 reviews related work on the Ceph cloud storage system and software-defined network and analyzes the limitations of the mapping algorithm in Ceph. In Section 3, we propose an improved Ceph-SDN framework and the scheme for monitoring and selecting modules. In Section 4, we set up a test environment to evaluate and verify the improvements of our node selection method.

2. **Related Works.**

The Ceph distributed storage system, is well-known in community applications and scientific research. Many enterprises have done significant work related to distributed data storage in both theoretical and practical ways. Dating back to 2008, the year when Ceph was first released, researchers such as Esteban Molina-Estolano and many others have developed some fundamental methods in regard to processing the Ceph read/write load, including weight balancing with a primary and secondary switch, which has since provided the basis for later research [9]. There has recently been much research interest related to Ceph loading. Sha. Edwin and others developed a divide-and-conquer strategy, which is based on MapReduce to ensure the balancing of load over storage nodes, which also solves the problem of computing migration when a system is faced with large volumes of data [10]. Michael Sevilla and others have also proposed a Ceph oriented load processing strategy [11]. Due to the open source character of Ceph, it is possible to optimize the node selection algorithm with multiple factors. This depends on obtaining the current network and host conditions.

Traditional methods of measuring network and flow data require significant configuration and instrumentation costs. A flexible and convenient method of obtaining network information is required to optimize the node selection process in a Ceph distributed storage system. As a new network management mode, SDN uses separate control and data planes and the unified management of the control layer. The SDN controller can monitor the global network state, the static topology, and the dynamic flow table. The SDN switch then only needs to perform the appropriate action according to the directions issued by the controller. Additionally, the software-defined network also provides the facility of network programming. The upper layer application calls the northbound interface to obtain network resources and services. The application of software-defined network in cloud storage and data centers has become a hot topic of research. Lai X and others developed a demand based SDN dynamic cloud storage service, adjusting the SDN controller domain by SDN dynamic regulation and reducing the time delay from the control layer to the data layer, thus improving the transfer efficiency of data storage. This has provided useful theoretical and practical support for the application of SDN in a cloud storage environment [12]. In conclusion, it is possible to combine SDN with the Ceph.

In a traditional network architecture, the measurement of network status requires complex configuration, and substantial measurement costs. The concept of SDN has inspired the proposal of many new methods of network measurement. Methods based on the SDN OpenFlow protocol [13] include Open TM [14], OpenNetMon [15], and Payless [16]. Methods built outside OpenFlow are OpenSketch [17], OpenSample [18], Devoflow [19], and PLANCK [20]. In general, OpenFlow compliance engenders least network cost, thus we use OpenFlow based traffic measurement combined with a Ryu controller [21] to obtain the network information and load status of the Ceph cluster. Ryu is a Python-based open source SDN controller, with a rich library of functionality that endows it with high development efficiency.

Multi-attribute decision-making addresses the problem of finite scheme decision making with multiple attributes. Some researchers have proposed that the number of decision items [22] in the interval number is represented by the number of areas. The probabilities of each value between the upper and lower limits are considered equal. However, when a variable is expressed by the interval number, the interval may be too small to cover the entire range of values [23]. To address this issue, [24] proposes to use triangular fuzzy numbers to represent the attribute values, so that the value range of the variable is maintained and the maximum possible value of the parameter can be identified. In [25], a Ceph node selection scheme combined with SDN is purposed. The multi-attribute problem is formulated by the TOPSIS [26] algorithm but it ignores the range over which the factors fluctuate over time as well as some other essential factors from both link and hosts. To objectively and accurately reflect the influence of multiple factors, we use a triangular fuzzy multi-attribute decision method based on grey relational analysis to weight the various networks in this paper. This decision-making algorithm can construct the ideal solution for decision-making problems.

3. Problem Statement and Preliminaries.

3.1. The data storage process of Ceph.

The excellent performance of the Ceph cloud storage system depends on the support of Reliable Autonomic Distributed Object Store (RADOS) [27]. RADOS contains object storage devices (OSDs) and a few monitor nodes which save mapping relations. An instance of the saving process can be divided into three main stages. In the first stage, files in the client would be divided into smaller objects on which RADOS can operate. Every object has a unique object ID formed by the node number and an object number. In the second stage, the concept of a placement group (PG) is introduced. It maps every object independently to PGs through a hashing algorithm. In the third stage, PGs, as the logical organization element of objects, would be duplicates according to the number of transcripts set by the user, saved into homologous OSDs through the CRUSH algorithm. The storage process is shown in Figure 1.

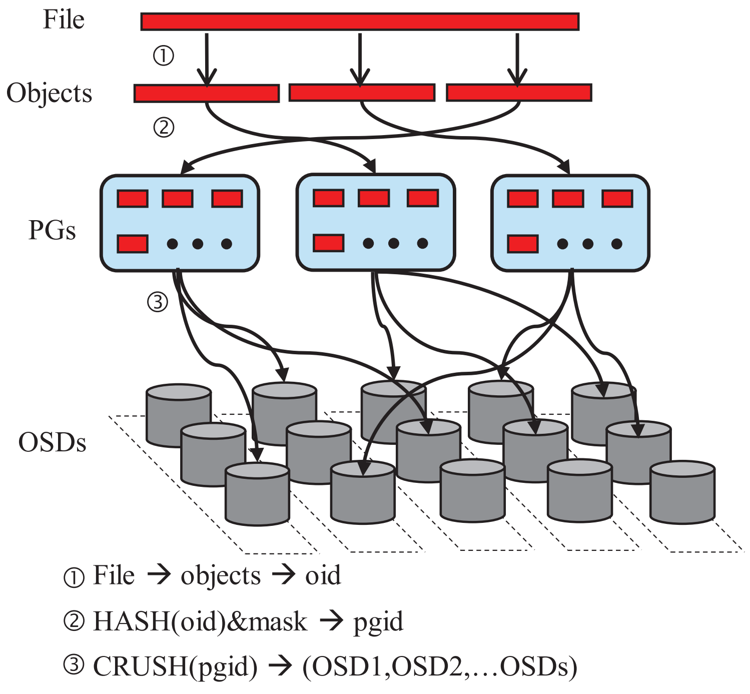


FIGURE 1. The storage process of data in the Ceph storage system

3.2. Limitations of the CRUSH algorithm in the storage process.

Taking the three-copies pattern as an example, the OSD with the highest performance is used as the primary OSD, the other OSDs will be assigned as the secondary OSD, the tertiary OSD, etc. The client will communicate directly with the primary OSD so that both read and write requests will be sent directly to the primary OSD. In the write process, the data is written to the primary OSD first, and then delivered to the other OSDs.

In contrast to a write-only request, the selection of primary OSD is more important in a read-only request, and it will directly affect the performance of the cluster. Moreover, every decision on allocating the primary or secondary OSD must be achieved through the CRUSH algorithm. However, the CRUSH algorithm in Ceph uses a weighting coefficient where the Ceph mechanism only considers the remaining capacity of the OSD as the determining factor in the weight coefficient. This is a restricted condition on which to achieve the purpose of balanced distribution of data.

By analyzing the source code of the Ceph cloud storage system and the performance of the storage clusters in actual operation, the quality of the link between storage nodes

OSDs and the loading of the OSD hosts themselves will also have a great impact on load balancing. As the node selection strategy in Ceph does not consider changes in the network link status and node load status, when these state factors deteriorate, the optimal OSD node may not be the one that was originally mapped. If the storage node is still selected based on its capacity, a capacity sufficient storage node with poor network performance or poor host status may become a bottle neck for the cluster.

In summary, the limitations of the original CRUSH algorithm of the Ceph storage cluster can lead to a decline in the performance of the cluster as a whole. In this paper, we propose a node selection method for a Ceph storage system based on software-defined network and a fuzzy multi-attribute decision method to improve the storage efficiency under the consideration of multiple network environments and host performance factors.

4. **Improved node selection method for a Ceph cloud storage system.** In this section we propose the system architecture first, then introduce the link and load information which is used as fuzzy multi-attribute model attribute parameters. To achieve the implementation of the parameters requirement system architecture, we propose the monitoring module and selection module, simultaneously.

4.1. **System architecture.** The system architecture is based on an SDN and Ceph cluster to develop the modified cloud storage system. In the architecture (shown in Figure 2), the Ceph cloud storage system is composed of a monitor node (MON), and a set of data storage nodes. The monitor node and data storage nodes are deployed within servers. The monitor node is responsible for monitoring all Ceph the storage nodes. Each storage node contains one or more OSDs and is directly connected to the cluster. The OpenFlow switches act as data transfer nodes to connect all servers and hosts. The monitoring module we propose is deployed in the SDN controller which gathers information on OSDs by sending query packets. The node selection module deployed in the Ceph monitor node remotely calls the information collected by the SDN controller to make the decision of selecting the optimal primary OSD.

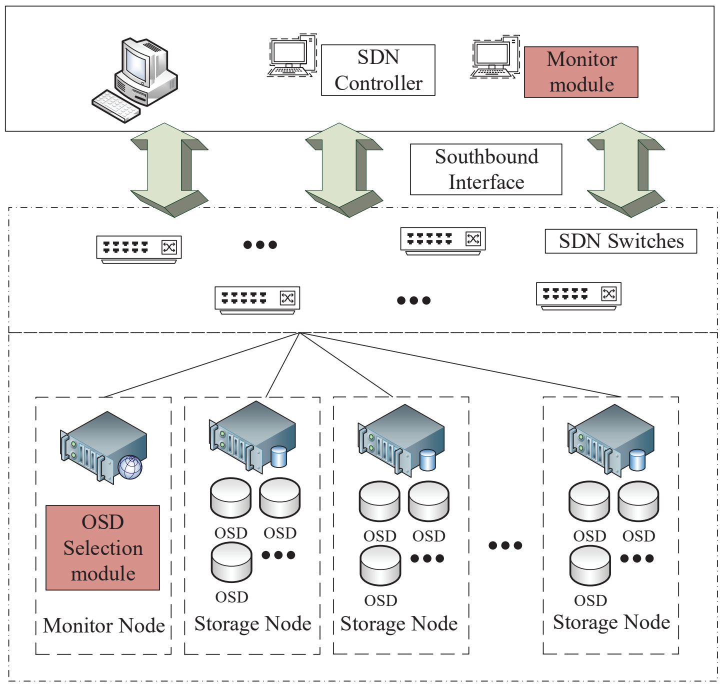


Figure 2. The system architecture

4.2. **The gathering of weight factors and the multi-attribute decision-making method.** Based on the architecture above, the implementation of the OSD nodes selection method requires whole cluster information as weight factors. This section will focus on the implementation of the selection algorithm in this SDN combining the Ceph distributed storage system.

4.2.1. *Determination of the OSD weight factors.* The improved Ceph storage system architecture takes the link quality factors and load factors of the storage nodes as the input for the modules. We choose the following characteristics of: network time delay, delay jitter, and remaining bandwidth as the link quality evaluation index. The I/O load, CPU utilization, and memory utilization are used to reflect the load states.

(1) Network time delay (D) and jitter (S): time delay is an important parameter for network performance which reflects the speed of communication between cluster nodes in the whole network. Jitter, meanwhile, is also one of the important factors affecting network QoS. It describes the degree of change (variation) in the group delay of packets. The factor is used to describe the degree of change in packet delay if there is congestion in the cluster network.

(2) Residual bandwidth of DSD nodes (B): the data transmission link of the network is analogous to a pipeline in engineering, and the residual bandwidth of OSD directly affects the read/write speed of the link between clients and OSD nodes.

(3) OSD utilization rate of the CPU (C) and memory utilization (R): The basic function of OSD requires resources supplied by memory and the CPU, so the CPU and memory performance have a direct effect on the performance of the OSD nodes.

(4) The I/O load of OSD (L) and OSD disk I/O load performance represent reading and writing performance. Considering the I/O load may lead to a more accurate selection of node weighting factors.

4.2.2. *The fuzzy multi-attribute decision method based OSD selection algorithm.* Facing with the fuzzy multi-attribute decision-making (FMADM) problem. We propose a triangle fuzzy multi-attribute decision based on grey correlation analysis [28] to balance capacity among storage nodes as well as setting the order of OSD nodes. The decision algorithm can construct the positive and negative ideal solution of the decision problem.

Apart from OSD storage capacity as defined in the CRUSH algorithm, we have identified six other factors that quantify OSD performance. We set A = {A1, A2, A3..., AN} as the OSD solution set, representing N OSD nodes. representing N OSD nodes. E = {E1, E2, E3, E4 E5 E6} is a set of attributes, E1-E6 respectively which represent network delay(D), delay jitter (S), memory utilization (R), CPU utilization (C), I/O load (L), and residual bandwidth (B). When the values of E1 to E5 increase, they will adversely affect the performance of the cluster. Therefore, E1, E2, E3, E4 and E5 are negative or cost attributes. As the value of E6 becomes larger, it will lead to an improvement in the performance of the cluster, so E6 (B) is a benefit type attribute. In this paper we propose a triangular fuzzy multi-attribute decision-making method based on a grey relational analysis method as follows:

Step 1: For solution set $A_j \in A$, factors $E_i \in E$ are to be measured and evaluated. Then we may obtain the attribute value of A_j and E_i as a set of triangular fuzzy numbers $r_{ij} = (a_{ij}^L, a_{ij}^M, a_{ij}^R)$ $(i = 1, 2, \ldots, 6; j = 1, 2, \ldots, N)$. Thus the decision matrix of triangular fuzzy numbers is then obtained:boxjoin() $R = (r_{ij})_{6*N}$ (Table 1). r_{ij}^M represents the value of the set of attributes that are remotely invoked by the monitoring node. r_{ij}^L and r_{ij}^R are the upper and lower limit values set according to the actual situation.

Step 2: Normalization of the decision matrix $R = (r)_{6*N}$ s carried out using Formula (1) and Formula (2). A new normalized decision matrix can be obtained $Z = (z_{ij})_{6*N}$ (Table 2).

TABLE 1. The decision matrix $R = (r_{ij})_{6*N}$

	A_1	A_2	A_3	...	A_N
E1	$(r_{11}^L, r_{11}^M, r_{11}^R)$	$(r_{12}^L, r_{12}^M, r_{12}^R)$	$(r_{13}^L, r_{13}^M, r_{13}^R)$	...	$(r_{1n}^L, r_{1n}^M, r_{1n}^R)$
E2	$(r_{21}^L, r_{21}^M, r_{21}^R)$	$(r_{22}^L, r_{22}^M, r_{22}^R)$	$(r_{23}^L, r_{23}^M, r_{23}^R)$	...	$(r_{2n}^L, r_{2n}^M, r_{2n}^R)$
E3	$(r_{31}^L, r_{31}^M, r_{31}^R)$	$(r_{32}^L, r_{32}^M, r_{32}^R)$	$(r_{33}^L, r_{33}^M, r_{33}^R)$	...	$(r_{3n}^L, r_{3n}^M, r_{3n}^R)$
...	...	...	...	...	
E6	$(r_{61}^L, r_{61}^M, r_{61}^R)$	$(r_{62}^L, r_{62}^M, r_{62}^R)$	$(r_{63}^L, r_{63}^M, r_{63}^R)$	...	$(r_{6n}^L, r_{6n}^M, r_{6n}^R)$

TABLE 2. The normalized decision matrix $Z = (z_{ij})_{6*N}$

	A_1	A_2	A_3	...	A_N
E1	$(z_{11}^L, z_{11}^M, z_{11}^R)$	$(z_{12}^L, z_{12}^M, z_{12}^R)$	$(z_{13}^L, z_{13}^M, z_{13}^R)$	...	$(z_{1n}^L, z_{1n}^M, z_{1n}^R)$
E2	$(z_{21}^L, z_{21}^M, z_{21}^R)$	$(z_{22}^L, z_{22}^M, z_{22}^R)$	$(z_{23}^L, z_{23}^M, z_{23}^R)$	...	$(z_{2n}^L, z_{2n}^M, z_{2n}^R)$
E3	$(z_{31}^L, z_{31}^M, z_{31}^R)$	$(z_{32}^L, z_{32}^M, z_{32}^R)$	$(r_{33}^{L}{}_{,33}^{M}{}_{,33}^{R})$	...	$(z_{3n}^L, z_{3n}^M, z_{3n}^R)$
...	...	...	...	...	
E6	$(z_{61}^L, z_{61}^M, z_{61}^R)$	$(z_{62}^L, z_{62}^M, z_{62}^R)$	$(z_{63}^L, z_{63}^M, z_{63}^R)$	...	$(z_{6n}^L, z_{6n}^M, z_{6n}^R)$

The formula for calculating the two required attributes is as follows:
Benefit attribute:

$$\begin{cases} Z_{ij}^L = \dfrac{r_{ij}^L}{\sqrt{\sum_{j=1}^n \exp(r_{ij}^R)}} \\ Z_{ij}^M = \dfrac{r_{ij}^M}{\sqrt{\sum_{j=1}^n \exp(r_{ij}^M)}} \\ Z_{ij}^R = \dfrac{r_{ij}^R}{\sqrt{\sum_{j=1}^n \exp(r_{ij}^L)}} \end{cases} \tag{1}$$

Cost attribute:

$$\begin{cases} Z_{ij}^L = \dfrac{(1/r_{ij}^R)}{\sqrt{\sum_{j=1}^n \exp(r_{ij}^L)}} \\ Z_{ij}^M = \dfrac{(1/r_{ij}^M)}{\sqrt{\sum_{j=1}^n \exp(r_{ij}^M)}} \\ Z_{ij}^R = \dfrac{(1/r_{ij}^L)}{\sqrt{\sum_{j=1}^n \exp(r_{ij}^R)}} \end{cases} \tag{2}$$

Step3: Determine the positive and negative ideal points.
The positive ideal point is defined by:

$$\begin{cases} \tilde{r}^{+} = \left[(z_1^{L+}, z_1^{M+}, z_1^{R+}), (z_2^{L+}, z_2^{M+}, z_2^{R+}), \ldots, (z_6^{L+}, z_6^{M+}, z_6^{R+})\right] \\ \tilde{r}^{-} = \left[(z_1^{L-}, z_1^{M-}, z_1^{R-}), (z_2^{L-}, z_2^{M-}, z_2^{R-}), \ldots, (z_6^{L-}, z_6^{M-}, z_6^{R-})\right] \end{cases} \tag{3}$$

z_i^+ & z_i^- is defined by:

$$\begin{cases} \tilde{z}_i^{+} = (z_i^{L+}, z_i^{M+}, z_i^{R+}) = \left[\max_j z_{ij}^L, \max_j z_{ij}^M, \max_j z_{ij}^R\right] \\ \tilde{z}_i^{-} = (z_i^{L-}, z_i^{M-}, z_i^{R-}) = \left[\min_j z_{ij}^L, \min_j z_{ij}^M, \min_j z_{ij}^R\right] \end{cases} \tag{4}$$

Step4: β is a multi-attribute resolution coefficient. In general, $\beta \in [0, 1]$. The trigonometric fuzzy grey correlation coefficient of each scheme and positive and negative ideal points is calculated as follows:

$$\begin{cases} \zeta_{ij}^{+} = \dfrac{\min\limits_{1\le i\le 6}\min\limits_{1\le j\le n} d(\tilde{r}_{ij}, r_j^+) + \beta \max\limits_{1\le i\le 6}\max\limits_{1\le j\le n} d(\tilde{r}_{ij}, r_j^+)}{d(\tilde{r}_{ij}, r_i^+) + \beta \max\limits_{1\le i\le 6}\max\limits_{1\le j\le n} d(\tilde{r}_{ij}, r_j^+)} \\ \zeta_{ij}^{-} = \dfrac{\min\limits_{1\le i\le 6}\min\limits_{1\le j\le n} d(\tilde{r}_{ij}, r_j^-) + \beta \max\limits_{1\le i\le 6}\max\limits_{1\le j\le n} d(\tilde{r}_{ij}, r_j^-)}{d(\tilde{r}_{ij}, r_i^-) + \beta \max\limits_{1\le i\le 6}\max\limits_{1\le j\le n} d(\tilde{r}_{ij}, r_j^-)} \end{cases} \tag{5}$$

$d(\tilde{a}, \tilde{b})$ is defined as:

$$d(\tilde{a}, \tilde{b}) = \sqrt{\frac{1}{3}[\exp(a^L - b^L) + \exp(a^M - b^M) + \exp(a^R - b^R)]} \tag{6}$$

Step5: $\omega = (\omega_1, \omega_2, \omega_3, \ldots, \omega_6)^{\mathrm{T}}$ is a weight vector for the attribute set E. $\omega_i \in [0, 1]$.

$$\sum_{i=1}^{6} \omega_i = 1 \tag{7}$$

The setting of weight ω_i is based on several experiments. According to the adjustment and deployment of the decision schemes under different weights, we found that the parameters of network delay and remaining bandwidth, have the greatest impact on decision making in the network state of our experimental environment. Hence, the weight of these two attributes was increased. For a different network and host state, many experiments would be needed to ensure that the weights of different networks are reasonable and appropriate.

$$\omega = (\omega_1, \omega_2, \omega_3, \ldots, \omega_6)^{\mathrm{T}} = [0.3, 0.08, 0.08, 0.08, 0.08, 0.3]^{\mathrm{T}} \tag{8}$$

The grey correlation degree of each scheme for positive and negative ideal points can finally be calculated by:

$$\begin{cases} \zeta_j^{+} = \dfrac{1}{m} \sum \omega_i \zeta_{ij}^{+}, j = 1, 2, \ldots, 6 \\ \zeta_j^{-} = \dfrac{1}{m} \sum \omega_i \zeta_{ij}^{-}, j = 1, 2, \ldots, 6 \end{cases} \tag{9}$$

Step 6: The grey comprehensive correlation of each storage node is calculated according to:

$$\varphi_j = \alpha_1 \zeta_j^{+} + \alpha_2 (1 - \zeta_j^{-}) \tag{10}$$

When the grey comprehensive correlation φ_j is larger, the OSD is more suitable for storage.

4.3. **Design and implementation of the monitoring module.** To achieve the fuzzy multi-attribute decision-making solutions in the previous section, the network status and several factors of loading are collected in the system mentioned above. In this section we implement the corresponding network host status monitoring module, which is divided into active and passive monitoring.

In active monitoring, the Ryu controller will send the query information to the OSD nodes after the module is built. Then, the SDN switches will unpack the returned packet to get the capacity of the port from the reply message. In passive mode, the SDN controller may receive packages which were delivered by OSD nodes to determine the load information of storage hosts. Each component is described in more detail in the next two sub-sections.

4.3.1. *Active monitoring module.* The Ryu controller will send the query information to the OSD nodes after the module is built. In the11 , SDN switches can obtain the link factors by unpacking the packets returned by the storage nodes. The information collected by the SDN switches contains the delivered pack numbers (tx), the number of receiving packets (rx) and the time interval (T) (duration) of data collection. In this paper, it contains the total bandwidth of the port (cs) and the formula for the residual bandwidth is as follows.

$$\text{Remaining Bandwidth} = \text{cs} - \frac{tx + rx}{T} \tag{11}$$

In the active process, we can gather packets through the Ryu controller. The gett_protocols() function can parse the packets to obtain the corresponding hosts IP and MAC address. The next step is to write the hosts IP information as well as the corresponding MAC into the dictionary ipt_mac. Meanwhile, each IP address and its corresponding port bandwidth is written to the ipt_bw dictionary. Then, the IP address is bound with the other message obtained in the active process.

4.3.2. *Passive monitoring module.* When an OpenFlow switch receives a data packet, the data matching field in the packet is extracted to map the received packets to the flow tables and find out where to deliver the packet. Each flow table must contain a stream table which can deal with a table-miss state. When the flow table does not match the corresponding flow table item, it is defined as a table-miss state, and the packet with the table-miss state will be sent to the Ryu controller.

OpenFlow switches, use the read function psutil() from Python to obtain the load status (disk I/O load, CPU utilization, memory utilization) when the read/write process is executing. The passive monitoring module sends the ICMP echo request to the storage nodes first, then it can get a receive packet with the link status. According to the returned link information (time delay, delay jitter), the five factors are packed into a UDP packet as well as creating a client socket and UDP connections. The next step is to periodically send UDP packets with messages to an IP address which does not exist in the flow table. According to the characteristics of the OpenFlow protocol, this packet will be sent straight to the Ryu controller. After the Ryu controller has executed packet parsing, we may obtain the source port which is located on the OSD nodes. Meanwhile, the parsing of the data section helps the Ryu controller find the load status of OSD nodes. The pseudo-code of the Ryu (SDN) controller (UDP server side) is shown in Table 3.

As the active monitoring module has matched the IP and port with the link bandwidth, the rest of the link factors and storage nodes host information will be mapped with an IP and port in the passive monitoring module.

Table 3. The pseudo code for Ryu controller gathering information

```
input: Packet_in packet
output: The host load status.
1:  @set_ev_cls(ofp_event.EventOFPPacketIn, MAIN_DISPATCHER)
2:  procedure: _packet_in_handler(ev)
3:     msg ← ev.msg
4:     pkt ← packet.Packet(msg.data)
5:     arp_header ← pkt.get_protocols(arp.arp) //address protocol
6:     ip_header ← pkt.get_protocol(ipv4.ipv4) //Ipv4packet protocol
7:     udp_header ← pkt.get_protocol(udp.udp) //Udp packet protocol
8:     if (ip_header and udp_header and udp_header.dst_port == xxxx)
       //The port and IP address which dose not exist in the cluster
9:        procedure: _parse_udp(msg.data)
10: end procedure
11: procedure: _parse_udp(msg.data)
12:    eth_data ← ethernet.ethernet.parser(msg.data)[2] // Resolving ports
13:    ip_data ← ipv4.ipv4.parser(eth_data)[2] //Resolving IP address
14:    udp_data ← udp.udp.parser(ip_data)[2] // Resolving UDP packet
15: end procedure
```

Table 4. The pseudo code of OSD selection

```
Input: OSD_INFO Map
Output: primary/secondary OSDs
1:  osd_addr, osd_value={};
2:  osd_addr = get_osd_addr();//IP of OSD hosts
3:  if osd is 'down' then
4:     remove osd from osd_addr;
5:  end
6:  osd_value[osd] = FMADM(OSD_INFO Map);
       // The fuzzy multi-attribute model is used to calculate the OSD's associated $\varphi_j$
7:  for osd in osd_addr and osd in osd_value do
8:     osd_primary_affinity = osd_value[osd]
9:     update {osd:osd_primary_affinity }
10: end
```

4.4. **OSD Selection Module.** To improve read efficiency and load balancing, the selection of storage nodes for the primary and secondary OSD processes is divided into two parts. First, the Ceph storage system selects a set of OSD nodes by storage capacity as a parameter. Second, after all the PGs have been mapped with OSD nodes, we calculate using the obtained value to obtain a new set of weights. Finally, we adjust the primary affinity of the OSD using the weight which is defined in the range of [0,1]. The primary affinity is an essential feature which determines whether an OSD node can serve as the main OSD. The pseudo-code for the program algorithm is shown in Table 4.

The primary OSD selection module is deployed in the Ceph monitor node. The module is mainly to used combine the cluster map of Ceph with the OSD map, which records the residual bandwidth, delay jitter, I/O load, CPU utilization, and memory utilization information of all OSD nodes. Periodically, the OSD map will be updated after a series of remote calls to the SDN controller. In the second stage of choosing the primary and secondary OSD nodes, the next issue to deal with is to adjust the original OSD weight according to the information in the OSD map. Finally, the optimal storage node selection scheme can be obtained through two weight status combinations.

5. **Experiment and result analysis.**

5.1. **Experiment environment.**

In this section, we will verify the read and write performance of the SDN based Ceph storage system in a real experimental environment. The Ceph cluster consists of five physical machines, including one monitoring node and four data storage OSD nodes where each OSD node contains three OSDs. The specific hardware environment is shown in Table 5.

Table 5. Physical environment hardware configurations.

function	number	Host configurations
Ceph cluster	5	CPU: Intel(R) Core i5-4690v2@2.10GHz Memory: 8GB Network card: 1Gb/s
SDN controller	1	CPU: Intel(R)Core i7-6700 @3.60GHz Memory: 16GB Network card: 1Gb/s
SDN switches	2	Alcatel-LucentOS6860E-48

The software environment is as follows: Ceph version: 0.94.10; Ryu controller version: 4.8; OpenFlow protocol version: 1.3; operating system: Ubuntu 14.05-server; KVM version: 2.

To test the read and write performance of the system, we used different sizes of objects, 4KB, 128KB, 512KB, 1024KB and 4096KB. The comparison between the original Ceph cluster and the Ceph with SDN & fuzzy multi-attribute decision-making is implemented by measuring the throughput and operation time from submission to completion in read/write operations with various object sizes.

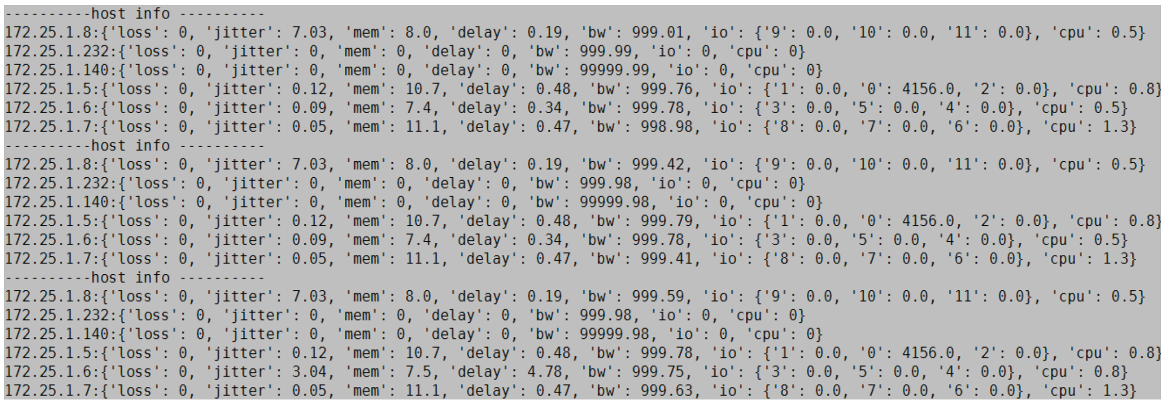
Upon entering the command "ryu-manger SDN_Monitor.py" on the host Ryu controller, the monitor program will be executed and the Ryu controller prints out the information obtained by the passive monitoring module in real time, as shown in Figure 3.

Here, 172.25.1.5, 172.25.1.6, 172.25.1.7 and 172.25.1.8 represent the IP addresses of the OSD hosts connected by OpenFlow switches. After the IP address, the six parameters: jitter, mem, delay, bw, io, cpu represent: delay jitter, memory utilization, link delay, residual bandwidth, I\O load, and CPU utilization.

5.2. **Results and Analysis.**

Figures 4 and 5 show the throughput of 100% write operations and 100% read operations under different workloads. The horizontal axis in the charts represents the file size (KB).

The design of the prototype selects the node with best performance to be the main OSD and the copy mode read operation only needs to read the primary OSD. As a result we found that the read operation has a significant improvement of 1520% for small files of less than 1024 KB. Furthermore, the throughput rate for three copies has nearly doubled when using the proposed method.



```
----------host info ----------
172.25.1.8:{'loss': 0, 'jitter': 7.03, 'mem': 8.0, 'delay': 0.19, 'bw': 999.01, 'io': {'9': 0.0, '10': 0.0, '11': 0.0}, 'cpu': 0.5}
172.25.1.232:{'loss': 0, 'jitter': 0, 'mem': 0, 'delay': 0, 'bw': 999.99, 'io': 0, 'cpu': 0}
172.25.1.140:{'loss': 0, 'jitter': 0, 'mem': 0, 'delay': 0, 'bw': 99999.99, 'io': 0, 'cpu': 0}
172.25.1.5:{'loss': 0, 'jitter': 0.12, 'mem': 10.7, 'delay': 0.48, 'bw': 999.76, 'io': {'1': 0.0, '0': 4156.0, '2': 0.0}, 'cpu': 0.8}
172.25.1.6:{'loss': 0, 'jitter': 0.09, 'mem': 7.4, 'delay': 0.34, 'bw': 999.78, 'io': {'3': 0.0, '5': 0.0, '4': 0.0}, 'cpu': 0.5}
172.25.1.7:{'loss': 0, 'jitter': 0.05, 'mem': 11.1, 'delay': 0.47, 'bw': 998.98, 'io': {'8': 0.0, '7': 0.0, '6': 0.0}, 'cpu': 1.3}
----------host info ----------
172.25.1.8:{'loss': 0, 'jitter': 7.03, 'mem': 8.0, 'delay': 0.19, 'bw': 999.42, 'io': {'9': 0.0, '10': 0.0, '11': 0.0}, 'cpu': 0.5}
172.25.1.232:{'loss': 0, 'jitter': 0, 'mem': 0, 'delay': 0, 'bw': 999.98, 'io': 0, 'cpu': 0}
172.25.1.140:{'loss': 0, 'jitter': 0, 'mem': 0, 'delay': 0, 'bw': 99999.98, 'io': 0, 'cpu': 0}
172.25.1.5:{'loss': 0, 'jitter': 0.12, 'mem': 10.7, 'delay': 0.48, 'bw': 999.79, 'io': {'1': 0.0, '0': 4156.0, '2': 0.0}, 'cpu': 0.8}
172.25.1.6:{'loss': 0, 'jitter': 0.09, 'mem': 7.4, 'delay': 0.34, 'bw': 999.78, 'io': {'3': 0.0, '5': 0.0, '4': 0.0}, 'cpu': 0.5}
172.25.1.7:{'loss': 0, 'jitter': 0.05, 'mem': 11.1, 'delay': 0.47, 'bw': 999.41, 'io': {'8': 0.0, '7': 0.0, '6': 0.0}, 'cpu': 1.3}
----------host info ----------
172.25.1.8:{'loss': 0, 'jitter': 7.03, 'mem': 8.0, 'delay': 0.19, 'bw': 999.59, 'io': {'9': 0.0, '10': 0.0, '11': 0.0}, 'cpu': 0.5}
172.25.1.232:{'loss': 0, 'jitter': 0, 'mem': 0, 'delay': 0, 'bw': 999.98, 'io': 0, 'cpu': 0}
172.25.1.140:{'loss': 0, 'jitter': 0, 'mem': 0, 'delay': 0, 'bw': 99999.98, 'io': 0, 'cpu': 0}
172.25.1.5:{'loss': 0, 'jitter': 0.12, 'mem': 10.7, 'delay': 0.48, 'bw': 999.78, 'io': {'1': 0.0, '0': 4156.0, '2': 0.0}, 'cpu': 0.8}
172.25.1.6:{'loss': 0, 'jitter': 3.04, 'mem': 7.5, 'delay': 4.78, 'bw': 999.75, 'io': {'3': 0.0, '5': 0.0, '4': 0.0}, 'cpu': 0.8}
172.25.1.7:{'loss': 0, 'jitter': 0.05, 'mem': 11.1, 'delay': 0.47, 'bw': 999.63, 'io': {'8': 0.0, '7': 0.0, '6': 0.0}, 'cpu': 1.3}
```

FIGURE 3. The factors gathered by Ryu controller

The model designed in this paper adopts the strategy of replica priority, so the bandwidth of the main OSD and other OSDs is consumed at the same time in writing operations. We found that the total bandwidth consumption in write operations was not significantly different from that of the original Ceph storage system. This shows that the scheme we proposed does not make a significant difference in optimizing the throughput of the write operation.

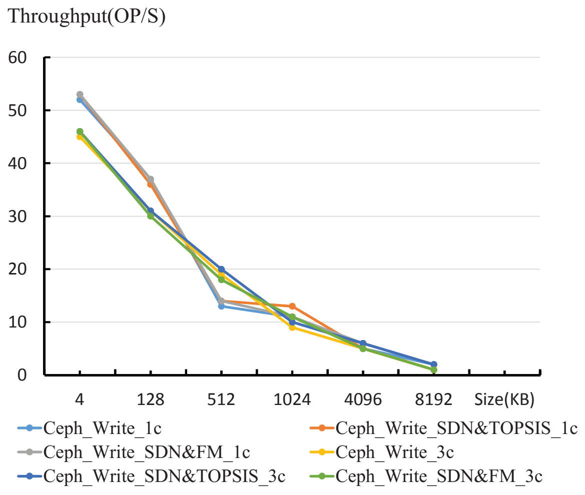


FIGURE 4. Comparison of 100% write operating throughput under different workload

Figures 6, 7, and 8 show the results of storage node response time for the improved model after storage and reading. To further explain the previous analysis, we note that the client only needs to access the primary OSD node in a read operation and selecting the optimal OSD as the primary OSD is based on the link quality and the underlying load selection can improve the efficiency of the client's read operation. Conversely, the write operation of the client needs all the data to be sent to the primary OSD, secondary OSD, and tertiary OSD, so the selection of which is the primary OSD has no effect on the write operation, resulting in no significant improvement.

It can be seen that the storage node selection scheme introduced in this paper reduces the response time of the 100% read operation for the 4KB object by 10ms (16.4%), and the response time of the 100% read operation of the 4096KB object is reduced by 120ms (18%). It is also demonstrated that the decision is made based on multiple weight factors. Further, fuzzy multi-attribute decision-making can achieve higher efficiency and performance.

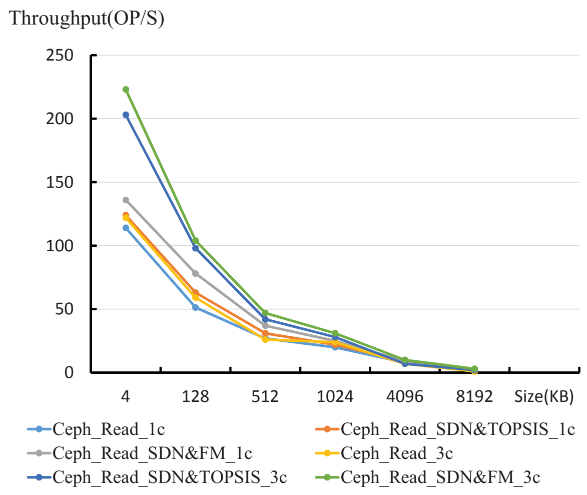


FIGURE 5. Comparison of 100% read operating throughput under different workload

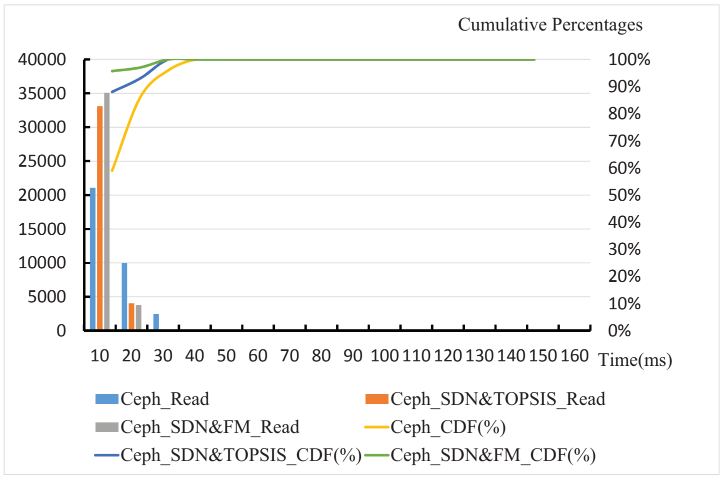


FIGURE 6. 100% read operation response time of 4KB_1c

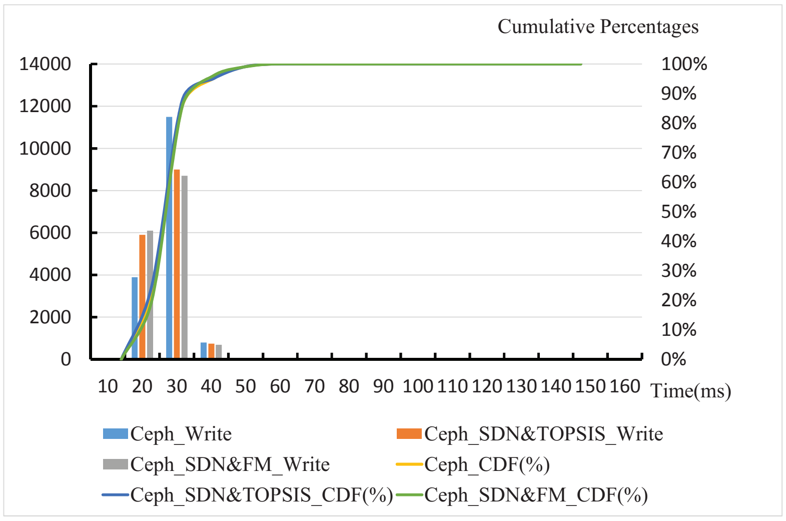


FIGURE 7. 100% read operation response time of 4096KB_1c

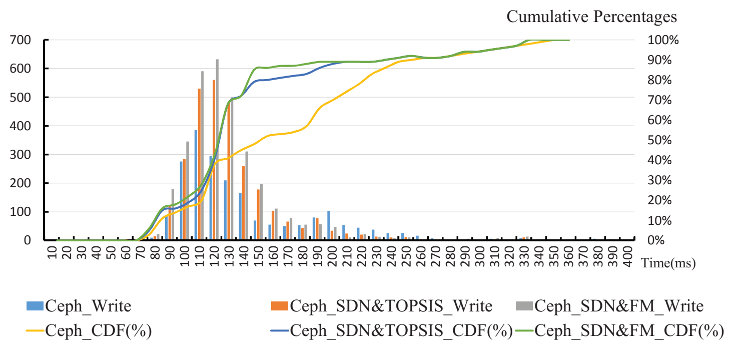


FIGURE 8. 100% write operation response time of 4KB_1c

The results show that the designed Ceph storage model combined with SDN improves the throughput of the small file reading operation and the response time of reading large files. However, there was no significant improvement in the write operation. The reason for this is that writing involves a series of operations in the write cache. The synchronous operation between the cache and the monitor consumes a large amount of I/O computing time, so the influence of bandwidth is no longer the dominant factor. Therefore, in write operations, the optimization scheme in this paper cannot significantly improve the write performance of the whole cluster. For the read operations, the operation only needs the OSD node with the minimum load (including the I/O load, CPU utilization, and bandwidth) to ensure the data transmission in the scheme we design. Hence, the performance of the read operation is significantly improved as expected.

6. Conclusion.

A node selection optimization method for the Ceph storage system, based on software-defined network and fuzzy a multi-attribute decision method is proposed is this paper. The improved method uses SDN technology to obtain the load status and link qualities of the storage nodes. It then solves the mathematical model of the multi parameter fuzzy multi-attribute decision-making matrix. By testing on a real platform, it is shown that the combination of SDN and Ceph can significantly improve the read performance of the Ceph storage system.

Acknowledgment. The work described in this paper was fully supported by The National Natural Science Foundation of China (No.61662018,61861013), China Postdoctoral Science Foundation (No.2016M602922XB), Guangxi Innovation-Driven Development Project (Major science and technology project No.AA18118031) ,The Natural Science Foundation of Guangxi Autonomous Region (No.2015GXNSFAA139294), Foundation of Guilin University of Technology (No.GUTQDJJ20172000019), Innovation Project of GUET Graduate Education grant from the Major State Basic Research Development (No. 2018YJCX53).

REFERENCES

[1] J M Menon, D A Pease, R M Rees, Distributed storage system for data-sharing among client computers running defferent operating system types: US, S20040122917[P]. 2004.
[2] S Ghemawat, H Gobioff, S T Leung, The google file system, *ACM SIGOPS operating systems review*, ACM, vo. 37, no. 5 pp. 29-43, 2003.
[3] S A Weil, S A Brandt, E L Miller, et al. Ceph: a scalable, high-performance distributed file system, *Symposium on Operating Systems Design and Implementation*, pp. 307-320, 2006.
[4] Y Zhang, S Debroy, P Calyam, Network measurement recommendations for performance bottleneck correlation analysis, *IEEE International Symposium on Local and Metropolitan Area Networks*, pp. 1-7, 2016.

[5] R G Clegg, M S Withall, A W Moore, et al, Challenges in the capture and dissemination of measurements from high-speed networks, *Communications Iet*, vol. 3, no. 6 pp. 957-966, 2009.

[6] P Sun, M Yu, M J Freedman, et al, HONE: joint host-network traffic management in software-defined networks, *Journal of Network & Systems Management*, vol. 23, no. 2, pp. 374-399, 2015.

[7] S Schaller, D Hood, Software-defined networking architecture standardization, *Computer Standards & Interfaces*, 54, 2017.

[8] O Kulak, C Kahraman, Fuzzy multi-attribute selection among transportation companies using axiomatic design and analytic hierarchy process, *Elsevier Science Inc*, 2005.

[9] E Molina-Estolano, C Maltzahn, S Brandt, Dynamic load balancing in ceph, 2008.

[10] E H M Sha, Y Liang, W Jiang, et al, Optimizing data placement of mapreduce on ceph-based framework under load-balancing constraint, *IEEE, International Conference on Parallel and Distributed Systems*, pp. 585-592, 2017.

[11] M A Sevilla, N Watkins, C Maltzahn, et al, Mantle: a programmable metadata load balancer for the ceph file system, *International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1-12, 2015.

[12] H Ma, Research on key technology of network service based on SDN in cloud environment, Beijing university of science and technology, 2016.

[13] X Lai, D Peng, D Huang, et al. Research on routing algorithm of data center network based on SDN, *Wireless Internet Technology*, 2016.

[14] N McKeown, T Anderson, H Balakrishnan, et al, OpenFlow: enabling innovation in campus networks, *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69-74, 2008.

[15] N L M van Adrichem, C Doerr, F A Kuipers, Opennetmon: network monitoring in openflow software-defined networks, *2014 IEEE Network Operations and Management Symposium*, pp. 1-8, 2014.

[16] S R Chowdhury, M F Bari, R Ahmed R, et al, PayLess: a low cost network monitoring framework for software defined networks, *2014 IEEE Network Operations and Management Symposium*, pp. 1-9, 2014.

[17] M Yu, L Jose, R Miao, Software defined traffic measurement with openSketch, *Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation*, pp. 29-42, 2013.

[18] J Suh, T T Kwon, C Dixon, et al, Opensample: a low-latency, sampling-based measurement platform for commodity SDN, *2014 IEEE 34th International Conference on Distributed Computing Systems*, pp. 228-237, 2014.

[19] A R Curtis, J C Mogul, J Tourrilhes, et al. Devoflow: scaling flow management for high-performance networks, *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 4, pp. 254-265, 2011.

[20] J Rasley, B Stephens, C Dixon, et al, Planck: millisecond-scale monitoring and control for commodity networks, *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 4, pp. 407-418, 2015.

[21] Ryu.RYU SDN Framework[EB/OL], https://osrg.github.io/ryu-book/en/html, 2017.

[22] Z S Xu, Q L Da, An uncertain ordered weighted geometric (UOWG) operator and its application, *Information: International Journal*, vol. 7, no. 2, pp. 175-182, 2004.

[23] G R Jahanshahloo, F Hosseinzadeh Lotfi, M Izadikhah, An algorithmic method to extend TOPSIS for decision-making problems with interval data, *Applied Mathematics and Computation*, vol. 176, no.2, pp. 1375-1384, 2006.

[24] GZ Bu, YW Zhang, Grey fuzzy comprehensive evaluation based on interval numbers of three parameters, *Systems Engineering and Electronics*, vol. 23, no. 9, pp. 43-45, 2001.

[25] D Wu, Y Wang, H Feng, et al. Optimization design and realization of ceph storage system based on software defined network, *International Conference on Computational Intelligence and Security. IEEE Computer Society*, pp. 277-281, 2017.

[26] H S Shih, H J Shyur, E S Lee, An extension of TOPSIS for group decision making, *Mathematical & Computer Modelling*, vol. 45, no. 7, pp. 801-813, 2007.

[27] S A Weil A W Leung, S A Brandt, et al. Rados: a scalable, reliable storage service for petabyte-scale storage clusters, *Proceedings of the 2nd international workshop on Petascale data storage: held in conjunction with Supercomputing'07*, pp. 35-44, 2007.

[28] G Wei, G Lan, Grey relational analysis method for interval-valued intuitionistic fuzzy multiple attribute decision making, *Fifth International Conference on Fuzzy Systems and Knowledge Discovery. IEEE Computer Society*, pp. 291-295, 2008.