A HRFDC strategy based on dynamic classification of failed cloud tasks

Bin Lian
Xi’an Shiyou University

Junqing Bai (✉️ 1363604262@163.com)
Xi’an Shiyou University

Research Article

Keywords: Cloud Data Centers, Fault-tolerant Strategy, Dynamic Classification, Rescheduling Algorithm

Posted Date: November 11th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-2236189/v1

License: © 国 This work is licensed under a Creative Commons Attribution 4.0 International License. Read Full License
Abstract

With the continuous development and improvement of cloud computing technology, the major computer giants have deployed their own cloud data center (CDC). At the same time, as user demands continue to expand, competition among cloud service providers is also intensifying. In order to continuously improve its own service quality and user satisfaction, cloud service providers adopting efficient and low-cost fault-tolerant strategy will improve the performance and profit of CDCs. However, the existing rescheduling strategy are mostly at the expense of the completion time of cloud task (CT) or increasing the compensation of cloud service providers, which ultimately leads to a decline in the profit of cloud service providers. More serious will affect the reputation and user experience of the enterprise. This paper systematically analyzes the performance loss caused by virtual machine (VM) failure and the rescheduling process of CDCs fault-tolerant strategy. At the same time, we established a dynamic classification rule of failed cloud task (FCT) according to the deadline for CTs. After that, a high-profit rescheduling fault-tolerant strategy for CDCs based on dynamic classification of FCTs (HRFDC) was proposed. This scheduling strategy maximizes the profitability of cloud service providers by increasing the failure repair rate of CDCs and reducing the compensation of cloud service providers. Finally, this strategy has been tested and verified, and its effect is due to the comparison algorithm.

1 Introduction

As more and more companies continue to increase investment in CDCs, large cloud service providers have multiple CDCs worldwide. For example, Amazon has deployed a total of 23 CDCs around the world, which covers the Americas, Asia, Europe and Africa. This will further intensify competition among cloud service providers. Therefore, improving the stability of the CDC can not only improve the user experience, but also increase the revenue of cloud service providers.

The fault-tolerant strategy for VM failure can be divided into copy replication and rescheduling. Copy replication generally begins at the initial stage of CT mapping. When the CDC scheduling system performs CT mapping, some redundant VMs are activated to map part of the CTs multiple times in order to avoid VM failure. It increases the probability of successful completion of CTs by increasing the number of copies. Although this method is simple and convenient, there are many hidden dangers. First, because the failure of the VM is a random event. The scheduling system cannot predict the occurrence of failures. Therefore, the chance of hitting the fault is low. Secondly, redundant VMs bring the cost of cloud service providers to rise and reduce their benefits. Finally, the choice of the number of copies is also a complicated issue. Designers need to comprehensively consider multidisciplinary knowledge such as statistics and semiconductors. Therefore, the losses caused by its complexity will be far greater than the benefits of the problem itself. Previous scholars have conducted some research on the rescheduling system [1–4]. However, these algorithms do not consider the following issues. First, CDCs are mostly heterogeneous clusters composed of cheap machines in reality. Use virtualization technology to rent and sell these resources to cloud users in the form of VMs. These VMs can be targeted to perform compute-intensive or memory-intensive CTs. Secondly, the CDC will also provide VMs with CPU of different
frequency for cloud users to choose. When rescheduling a FCT, the system can select a VM with a high-frequency CPU to map based on its deadline to save costs. Thirdly, due to the abandonment of some CTs in the previous research, this means that ultimately the results cannot be delivered to cloud users. This will not only bring about a reduction in revenue, but also the loss of service brand and corporate image. Although the delayed delivery will reduce the chance of cloud users purchasing cloud services again, it at least guarantees the basic reputation of the enterprise. Therefore, even if there will be a large cost, the cloud service provider should complete all CTs as much as possible.

Based on the above reasons, we first analyzed the overall architecture and model of the CDC in this article. Secondly, the effect of CPU frequency in the VM on the rescheduling strategy is studied. Thirdly, a dynamic classification rule for FCTs is established according to the deadline of CTs. Then, a high-profit rescheduling fault-tolerance strategy for CDCs based on dynamic classification of FCTs (HRFDC) is proposed. Finally, the strategy is verified.

The main contributions of this paper are as follows:

(1) We establish a rescheduling CT and VM model.

(2) We determine the choice of rescheduling strategy for high-frequency VMs and computational VMs.

(3) We establish a dynamic classification rule for FCTs and propose the HRFDC strategy.

(4) We use the real VM parameters provided by Amazon to verify the effect of the HRFDC strategy.

Section 2 introduces the relevant work. Section 3 explains the CDC rescheduling model and rule. Section 4 is the HRFDC. The test results of the algorithm appear in Section 5. The full text is summarized in Section 6.

2 Related Work

The research is mainly divided into three aspects based on performance, energy consumption and profit.

2.1 Scheduling strategy based on performance

Qiu, et al. [5] proposed a reliability-based optimization framework ROCloud to improve the reliability of applications through fault tolerance. ROCloud contains two ranking algorithms. Zhao and Sakellariou [6] focus on providing fairness and four other strategies that can be used to arrange multiple DAGs are proposed and evaluated. Zhang, et al. [7] focus on workflow scheduling mechanism. Although the static scheduling method for workflow applications has a lot of work in a parallel environment, there is very little work done in an actual multi-cluster grid environment. Yu and Shi [8] proposed a novel adaptive rescheduling concept, which allows the workflow planning program and the runtime execution program to work in tandem and re-plan in a proactive manner when the grid environment changes significantly. Ostermann, et al. [9] analyzed two workflow-based workload tracking from the Austrian grid. They
introduced a method for analyzing such traces, focusing on the intrinsic characteristics of the workflow and the characteristics related to the environment. Ding, et al. [10] proposed an offline fault-tolerant elastic scheduling algorithm (FTESW) for cloud system workflow. Xiao, et al. [11] proposed a system that uses virtualization technology to dynamically allocate data center (DC) resources according to application requirements and supports green computing by optimizing the number of servers in use.

2.2 Scheduling strategy based on energy consumption

He, et al. [12] considered the scheme of hosting multiple VM clusters in a cloud system composed of physical node clusters. Wood, et al. [13] described the linear programming formula for static and dynamic server integration. Map VMs with unique attributes to a specific set of physical servers and limit the total number of dynamically merged migrations. Pinheiro, et al. [14] proposed a technology to minimize the power consumption of computing nodes serving multiple Web applications in a heterogeneous cluster. Chieu and Chan [15] proposed a centralized supply and expansion management system, which regularly performs maintenance services through performance and capacity data retrieved from each data monitoring agent at regular intervals, integrates VMs, and deletes idle physical hosts. Fei, et al. [16] introduced the concept of threshold-based load balancing in order to enhance performance measurement such as system utilization. Sharma, et al. [17] conducted a lot of research work on the thermal resource management of the DC. Guo and Fang [18] implemented and evaluated in distributed simulators on grid and cloud deployment, applying optimization techniques to solve the problems of optimal traffic distribution, server configuration, and battery management in the DC.

2.3 Scheduling strategy based on profit

Mao and Humphrey [19] proposed a method in which the basic computing elements are VMs of various sizes, specifying the work as a workflow. Topcuoglu, et al. [20] proposed two novel scheduling algorithms for a limited number of heterogeneous processors. The purpose is to meet the requirements of high performance and fast scheduling time. Chen, et al. [21] considered the energy efficiency management of homogeneous resources in Internet hosting. The main challenge is to determine the resource requirements such that each application is at its current request load level and allocates resources in the most efficient manner, which can be based on the available budget and Negotiate the service needs of current users, that is, balance the cost of resource use and benefits. Raghavendra, et al. [22] discussed from the perspective of control theory and a feedback control loop is applied to coordinate the actions of the controller by combining and managing the DC environment to coordinate five different power management policies. Dodonov and Mello [23] proposed a method to schedule distributed application communication activities in the grid based on predictions. Ardagna, et al. [24] introduced multi-standard decision-making technology and replaced the use of meta-heuristic algorithm. In order to avoid the problems of low resource sharing efficiency, low utilization rate, waste of resources and revenue loss caused by static allocation strategies and pricing models, the supplier shut down the PM to save energy and reduce costs.
In summary, the above scheduling strategy solves some research problems of performance, energy consumption and profit. However, the actual CDC scheduling system cannot fully guarantee the reliability of the PM. Large CDCs have certain fault-tolerant strategies. These fault-tolerant strategies can ensure the stability and continuity of the CDC. At the same time, the choice of fault-tolerant strategy is also a core issue. The rescheduling strategy is not particularly sensitive to cost. However, because rescheduling occurs after a failure of VM, it will have a great impact on deadline-sensitive CTs. In order to solve the CT timeout problem caused by rescheduling, we can speed up the execution of CTs by increasing the execution speed. A dynamic classification rule for FCTs is established for the deadline of CTs. Finally, the HRFDC strategy is proposed. This scheduling strategy maximizes the profitability of cloud service providers by increasing the failure repair rate of CTs and reducing the compensation of cloud service providers.

3 Cdc Rescheduling Model And Rule

3.1 CDC architecture

The CDC contains a set of VMs VM={vm$_1$,vm$_2$,...,vm$_n$}, where n represents the total number of VMs provided by the CDC. Each VM can be expressed as vm$_i$={c$_i$,m$_i$,s$_i$,p$_i$,n$_i$,t$_i$,e$_i$}. The c$_i$,m$_i$,s$_i$,p$_i$,n$_i$,t$_i$,e$_i$ represent the number of CPU cores, memory size (GB), processing speed of CPU cores (MIPS), the price of CPU cores included in each VM ($ / hour, low frequency = price\_low, high frequency = price\_high), the CT number running on the VM, the running time of the VM (s), and the state of the VM (1: normal, 2: failure and 3: copy replication). CTs submitted by cloud users can also be represented as a set of CTs. CT={ct$_1$, ct$_2$,..., ct$_m$}, where m represents the number of CTs. Each CT can be described as ct$_j$={l$_j$,m$_j$,d$_j$,t$_j$,p$_j$,f$_j$}, where l$_j$,m$_j$,d$_j$,t$_j$,p$_j$,f$_j$ represent the length of the CT (MI), the required memory size (GB), the deadline of the CT (s), the required actual time of the CT (s), the benefit of completing the CT ($), and the cost of failure ($).

The calculation method of the required actual time of CT j is as follows:

\[
ct.t_j = \frac{ct.l_j}{((f/1000) \times \text{vm.c}_i \times 0.9)}
\]

1

Where f is the CPU frequency of the VM.

CDCs generally have 10% redundancy due to fluctuations in the VM performance [25]. Therefore, the deadline for providing cloud users is 1.1 times the required actual time. The calculation method of the deadline for CT j is as follows:

\[
ct.d_j = 1.1 \times ct.t_j
\]

2

The benefit of completing CT j are as follows:
Cloud users need to rent VMs to complete the CTs in the CDC. Mature CDCs provide VMs with standard machine specifications for users to choose. For example, Amazon’s Eastern United States (Ohio) DC provides conventional VMs that contain four series. The parameters shown in Table 1 are the minimum models for each series. For example, m5a.large (Low-frequency Universal) is the lowest configuration of this series of VMs. At the same time, the CDC also provides high-frequency VM series m5n.large (High-frequency Universal) and m5n.large (High-frequency Computational). The configuration of these two series of VMs is only different in frequency compared to the configuration of low frequency. At the same time, the CDC also provides VMs corresponding to these four VMs with the same scale expansion. The m5a.large (Low-frequency Universal) doubled VM is m5a.xlarge (Low-frequency Universal). The m5a.24xlarge (Low-frequency Universal) configures the largest VM for this series. The number of CPU cores and the size of memory have been increased by 24 times. Of course, the rental price will also increase in proportion to the allocation.

<table>
<thead>
<tr>
<th>Table 1 price of VM (Ohio)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>frequency</strong></td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>m5a.large(Low-frequency Universal)</td>
</tr>
<tr>
<td>m5n.large(High-frequency Universal)</td>
</tr>
<tr>
<td>m5a.large(Low-frequency Computational)</td>
</tr>
<tr>
<td>m5n.large(High-frequency Computational)</td>
</tr>
</tbody>
</table>

We assume that cloud users submit a collection of CTs. CDCs need to map them. We only analyze the first five CTs, and their parameters are shown in Table 2. First, the smallest VM that meets the memory requirements is generally selected in order to meet the memory requirements of CTs and minimize the cost. We assume that all low-frequency Universal VMs are selected during the initial mapping. The CT ct₁ will choose the m5a.large (Low-frequency Universal) VM. We calculate the required actual time, deadline and benefit according to formulas 1–3, respectively. At the same time, other CTs are also deployed in the same way. The VM is shown in Fig. 1 after the initial mapping. Each cell represents a VM of m5a.large (Low-frequency Universal) specification in the figure. The parameters of the final five CTs are shown in Table 3. The parameters of the five corresponding VMs are shown in Table 4.
Table 2
parameters before CT mapping

<table>
<thead>
<tr>
<th>l</th>
<th>m</th>
</tr>
</thead>
<tbody>
<tr>
<td>ct₁</td>
<td>90000MI</td>
</tr>
<tr>
<td>ct₂</td>
<td>60000MI</td>
</tr>
<tr>
<td>ct₃</td>
<td>80000MI</td>
</tr>
<tr>
<td>ct₄</td>
<td>100000MI</td>
</tr>
<tr>
<td>ct₅</td>
<td>120000MI</td>
</tr>
</tbody>
</table>

Table 3
CT parameters after CT mapping

<table>
<thead>
<tr>
<th>l</th>
<th>m</th>
<th>d</th>
<th>t</th>
<th>p</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>ct₁</td>
<td>90000MI</td>
<td>8GB</td>
<td>22.0s</td>
<td>20.0s</td>
<td>(1.72/3600)$</td>
</tr>
<tr>
<td>ct₂</td>
<td>60000MI</td>
<td>16GB</td>
<td>7.4s</td>
<td>6.7s</td>
<td>(1.15/3600)$</td>
</tr>
<tr>
<td>ct₃</td>
<td>80000MI</td>
<td>16GB</td>
<td>9.7s</td>
<td>8.8s</td>
<td>(1.51/3600)$</td>
</tr>
<tr>
<td>ct₄</td>
<td>100000MI</td>
<td>8GB</td>
<td>24.2s</td>
<td>22.2s</td>
<td>(1.91/3600)$</td>
</tr>
<tr>
<td>ct₅</td>
<td>120000MI</td>
<td>8GB</td>
<td>29.3s</td>
<td>26.6s</td>
<td>(2.288/3600)$</td>
</tr>
</tbody>
</table>

Table 4
VM parameters after CT mapping

<table>
<thead>
<tr>
<th>c</th>
<th>m</th>
<th>s</th>
<th>p</th>
<th>n</th>
<th>t</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>vm₁</td>
<td>2</td>
<td>8GB</td>
<td>4500MIPS</td>
<td>0.086</td>
<td>USD/h</td>
<td>1</td>
</tr>
<tr>
<td>vm₂</td>
<td>4</td>
<td>16GB</td>
<td>9000MIPS</td>
<td>0.172</td>
<td>USD/h</td>
<td>2</td>
</tr>
<tr>
<td>vm₃</td>
<td>4</td>
<td>16GB</td>
<td>9000MIPS</td>
<td>0.172</td>
<td>USD/h</td>
<td>3</td>
</tr>
<tr>
<td>vm₄</td>
<td>2</td>
<td>8GB</td>
<td>4500MIPS</td>
<td>0.086</td>
<td>USD/h</td>
<td>4</td>
</tr>
<tr>
<td>vm₅</td>
<td>2</td>
<td>8GB</td>
<td>4500MIPS</td>
<td>0.086</td>
<td>USD/h</td>
<td>5</td>
</tr>
</tbody>
</table>

3.3 VM failure

Rescheduling technology is commonly used in modern CDCs. As a mature cloud service provider, it usually improves the performance of new VMs to ensure deadline. According to Amazon's service
agreement, cloud users will be compensated based on the downtime of VM. We can turn it into compensation beyond the deadline. The specific compensation ratio is shown in Table 5 according to the exceeding ratio the deadline.

<table>
<thead>
<tr>
<th>exceeding ratio</th>
<th>compensation ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0–1%)</td>
<td>10%</td>
</tr>
<tr>
<td>(1%-5%)</td>
<td>25%</td>
</tr>
<tr>
<td>exceed 5%</td>
<td>100%</td>
</tr>
</tbody>
</table>

### 3.4 CDC rescheduling

When a VM center fails in the CDC, its corresponding CT will fail. In order to continue to execute the FCT, the scheduling system needs to reschedule the CT and activate a new VM for mapping. If you activate a VM with the same configuration, the completion time of the FCT may exceed the deadline. This is especially obvious when the fault occurs late. This not only faces higher compensation but also affects the reputation of the cloud service provider. Therefore, the rescheduling system usually expands the VM configuration in the same proportion to ensure that CTs can be completed on time. However, this will also significantly increase the operating costs of cloud service providers. Through analysis of VMs provided by cloud service providers such as Amazon and Alibaba Cloud. Cloud users usually rent Low-frequency Universal VMs. We can also increase the execution speed by using high-frequency or computational VMs. We can see that their prices have risen in turn from Table 1. Based on this, we propose a dynamic classification rule for FCTs.

**Rule 1. Dynamic classification rule of FCTs.** When remapping FCTs, we can change the type of VM so that the FCT meets its deadline. If the original configured VM is used and the deadline is met, the FCT is classified as a Low-frequency Universal CT. If the frequency of the CPU needs to be increased to 3.1 GHz to meet the deadline, the FCT is classified as a High-frequency Universal CT. If the ratio of the number of CPU cores and the size of memory needs to be increased to 4:8 to meet the deadline, the FCT is classified as a Low-frequency Computational CT. If the two aspects need to be increased simultaneously to meet the deadline, the FCT is classified as a High-frequency Computational CT. If none of the above can be satisfied, expand the four configurations of VMs by the same proportion and select the VM with the lowest cost for classification and remapping.

We still use the previous example to demonstrate. Assume the time of the VM fails is random. Table 6 shows the remaining deadline for FCTs. Other parameters remain unchanged. According to the dynamic classification rule of FCTs, if ct₁ is mapped on the originally configured VM, it can meet the deadline requirement. The ct₁ is a Low-frequency Universal CT. If ct₂ is mapped on the originally configured VM, the deadline cannot be met. When the frequency of the CPU is increased to 3.1 GHz, it can meet the
deadline requirement. The \( \text{ct}_2 \) is a High-frequency Universal CT. The \( \text{ct}_3 \) needs to change the ratio of the number of CPU cores and the size of memory to 4:8 to meet the deadline. Therefore, \( \text{ct}_3 \) is a Low-frequency Computational CT. The same method can be concluded that \( \text{ct}_4 \) is a High-frequency Computational CT. The \( \text{ct}_5 \) wasted a lot of time due to the late failure of the VM. If the original configuration of the VM is still used for mapping, no matter how high the frequency or the ratio, the deadline cannot be met. We have to double the VM configuration and use High-frequency Computational VMs to meet the deadline. Therefore, the \( \text{ct}_5 \) is also a High-frequency Computational CT.

| \( \text{ct}_1 \) | 90000MI | 8GB | 20.0s | 20.0s | Low-frequency Universal |
| \( \text{ct}_2 \) | 60000MI | 16GB | 5.4s | 6.7s | High-frequency Universal |
| \( \text{ct}_3 \) | 80000MI | 16GB | 4.5s | 8.8s | Low-frequency Computational |
| \( \text{ct}_4 \) | 100000MI | 8GB | 9.3s | 22.2s | High-frequency Computational |
| \( \text{ct}_5 \) | 120000MI | 8GB | 5.4s | 26.6s | High-frequency Computational |

After the above analysis, we have classified the FCTs. After that, the corresponding VM is activated according to the required VM parameters for remapping. The VM parameters after rescheduling are shown in Table 7.
Table 7
VM parameters after rescheduling

<table>
<thead>
<tr>
<th></th>
<th>c</th>
<th>m</th>
<th>s</th>
<th>p</th>
<th>n</th>
<th>t</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>vm₁</td>
<td>2</td>
<td>8GB</td>
<td>4500MIPS</td>
<td>0.086 USD/h</td>
<td>1</td>
<td>2.0s</td>
<td>2</td>
</tr>
<tr>
<td>vm₂</td>
<td>4</td>
<td>16GB</td>
<td>9000MIPS</td>
<td>0.172 USD/h</td>
<td>2</td>
<td>2.0s</td>
<td>2</td>
</tr>
<tr>
<td>vm₃</td>
<td>4</td>
<td>16GB</td>
<td>9000MIPS</td>
<td>0.172 USD/h</td>
<td>3</td>
<td>5.2s</td>
<td>2</td>
</tr>
<tr>
<td>vm₄</td>
<td>2</td>
<td>8GB</td>
<td>4500MIPS</td>
<td>0.086 USD/h</td>
<td>4</td>
<td>14.9s</td>
<td>2</td>
</tr>
<tr>
<td>vm₅</td>
<td>2</td>
<td>8GB</td>
<td>4500MIPS</td>
<td>0.086 USD/h</td>
<td>5</td>
<td>23.9s</td>
<td>2</td>
</tr>
<tr>
<td>vm₆</td>
<td>2</td>
<td>8GB</td>
<td>4500MIPS</td>
<td>0.086 USD/h</td>
<td>1</td>
<td>20.0s</td>
<td>3</td>
</tr>
<tr>
<td>vm₇</td>
<td>4</td>
<td>16GB</td>
<td>11160MIPS</td>
<td>0.238 USD/h</td>
<td>2</td>
<td>5.3s</td>
<td>3</td>
</tr>
<tr>
<td>vm₈</td>
<td>8</td>
<td>16GB</td>
<td>18000MIPS</td>
<td>0.258 USD/h</td>
<td>3</td>
<td>4.4s</td>
<td>3</td>
</tr>
<tr>
<td>vm₉</td>
<td>4</td>
<td>8GB</td>
<td>11160MIPS</td>
<td>0.356 USD/h</td>
<td>4</td>
<td>8.9s</td>
<td>3</td>
</tr>
<tr>
<td>vm₁₀</td>
<td>8</td>
<td>16GB</td>
<td>22320MIPS</td>
<td>0.712 USD/h</td>
<td>5</td>
<td>5.3s</td>
<td>3</td>
</tr>
</tbody>
</table>

It can be seen from the above example that the five FCTs all meet the deadline requirements after remapping. At the same time, the dynamic classification rule of FCTs selects VMs reasonably according to the classification of CTs. This avoids blindly expanding the capacity of the VM. In the next part, the core HRFDC strategy of this article will be introduced.

4 Hrfdc Strategy

The first analysis of the overall architecture of the CDC. Secondly, we further analyzes the initialization process of CTs and the failure of VMs. Finally, the rescheduling process of the CDC is analyzed, and dynamic classification rule of FCTs are proposed to deal with the rescheduling problem of VM failures. Based on this idea, we propose the HRFDC strategy. This scheduling strategy maximizes the profitability of cloud service providers by increasing the failure repair rate of CTs and reducing the compensation of cloud service providers. According to the previous conclusion, the HRFDC strategy as shown in Strategy 1.
**Strategy 1: (HRFDC)**

**Input:** CT collection: CT={ct₁, ct₂, ..., ctₘ},
VM collection: VM={vm₁, vm₂, ..., vmₙ}

**Output:** VM collection: VM={vm₁, vm₂, ..., vmₙ}

1. while VM not null do
2. if (vm.fᵢ = 2) then
3. if ((ct.lᵢ/ct.dᵢ) ≤ (vm.cᵢ*2.5*1000*0.9)) then
4. VM = LFU(VM, ctᵢ)
5. end if
6. if ((vm.cᵢ*3.1*1000*0.9) ≥ (ct.lᵢ/ct.dᵢ) > (vm.cᵢ*2.5*1000*0.9)) then
7. VM = HFU(VM, ctᵢ)
8. end if
9. if ((vm.cᵢ*2*2.5*1000*0.9) ≥ (ct.lᵢ/ct.dᵢ) > (vm.cᵢ*3.1*1000*0.9)) then
10. VM = LFC(VM, ctᵢ)
11. end if
12. if ((vm.cᵢ*2*3.1*1000*0.9) ≥ (ct.lᵢ/ct.dᵢ) > (vm.cᵢ*2*3.1*1000*0.9)) then
13. VM = HFC(VM, ctᵢ)
14. end if
15. if ((ct.lᵢ/ct.dᵢ) > (vm.cᵢ*2*3.1*1000*0.9)) then
16. ct.lᵢ = ct.lᵢ/2ⁿ
17. end if
18. VM = HRFDC(VM, CT)
19. end while

First select the VM that caused the failure in the strategy (line 1–2). Second, dynamically classify the FCTs on the failed VM and call different strategies (line 3–19).

The Strategy 1 selects a Low-frequency Universal VMs for mapping (line 4). In this case, the original configured VM can meet the deadline. As shown in Strategy 2.
**Strategy 2: (LFU)**

| Input: Failed CT collection: $ct_i$,  
| VM collection: $VM=\{vm_1, vm_2, ..., vm_n\}$  
| Output: VM collection: $VM=\{vm_1, vm_2, ..., vm_n\}$  
| 1 $vm.c.(n+i) = vm.c_i$  
| 2 $vm.m.(n+i) = vm.m_i$  
| 3 $vm.s.(n+i) = vm.c.(n + i)*(2.5*1000*0.9)$  
| 4 $vm.p.(n+i)=(price_low/3600)* vm.c.(n + i)/2$  
| 5 $vm.n.(n+i) = i$  
| 6 $vm.t.(n+i) = ct.l./ (vm.s.(n+i) *2.5*1000*0.9)$  
| 7 $vm.e.(n+i) = 1$  

The Strategy 1 selects a High-frequency Universal VM for mapping (line 7). At this time, it increases the frequency of the CPU to 3.1 GHz can meet the deadline. As shown in Strategy 3.

**Strategy 3: (HFU)**

| Input: Failed CT collection: $ct_i$,  
| VM collection: $VM=\{vm_1, vm_2, ..., vm_n\}$  
| Output: VM collection: $VM=\{vm_1, vm_2, ..., vm_n\}$  
| 1 $vm.c.(n+i) = vm.c_i$  
| 2 $vm.m.(n+i) = vm.m_i$  
| 3 $vm.s.(n+i) = vm.c.(n + i)*(3.1*1000*0.9)$  
| 4 $vm.p.(n+i)=(price_high/3600)* vm.c.(n + i)/2$  
| 5 $vm.n.(n+i) = i$  
| 6 $vm.t.(n+i) = ct.l./ (vm.s.(n+i) *3.1*1000*0.9)$  
| 7 $vm.e.(n+i) = 1$  

The Strategy 1 selects a Low-frequency Computational VM for mapping (line 10). At this time, it increases the ratio of the number of CPU cores and the size of memory to 4: 8 can meet the deadline. As shown in
Strategy 4.

<table>
<thead>
<tr>
<th>Strategy 4: (LFC)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> Failed CT collection: ( ct_i ), VM collection: ( VM={vm_1,vm_2,...,vm_n} )</td>
</tr>
<tr>
<td><strong>Output:</strong> VM collection: ( VM={vm_1,vm_2,...,vm_n} )</td>
</tr>
<tr>
<td>1 ( vm.c_{(n+i)} = vm.c_i \times 2 )</td>
</tr>
<tr>
<td>2 ( vm.m_{(n+i)} = vm.m_i )</td>
</tr>
<tr>
<td>3 ( vm.s_{(n+i)} = vm.c.(n + i) \times (2.5 \times 1000 \times 0.9) )</td>
</tr>
<tr>
<td>4 ( vm.p_{(n+i)} = (price_low \times 1.5 / 3600) \times vm.c.(n + i) / 2 )</td>
</tr>
<tr>
<td>5 ( vm.n_{(n+i)} = i )</td>
</tr>
<tr>
<td>6 ( vm.t_{(n+i)} = ct.l_i / (vm.s_{(n+i)} \times 2.5 \times 1000 \times 0.9) )</td>
</tr>
<tr>
<td>7 ( vm.e_{(n+i)} = 1 )</td>
</tr>
</tbody>
</table>

The Strategy 1 selects a High-frequency Computational VM for mapping (line 13). At this time, the two aspects need to be increased at the same time to meet the deadline. As shown in Strategy 5.

<table>
<thead>
<tr>
<th>Strategy 5: (HFC)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> Failed CT collection: ( ct_i ), VM collection: ( VM={vm_1,vm_2,...,vm_n} )</td>
</tr>
<tr>
<td><strong>Output:</strong> VM collection: ( VM={vm_1,vm_2,...,vm_n} )</td>
</tr>
<tr>
<td>1 ( vm.c_{(n+i)} = vm.c_i \times 2 )</td>
</tr>
<tr>
<td>2 ( vm.m_{(n+i)} = vm.m_i )</td>
</tr>
<tr>
<td>3 ( vm.s_{(n+i)} = vm.c.(n + i) \times (2.5 \times 1000 \times 0.9) )</td>
</tr>
<tr>
<td>4 ( vm.p_{(n+i)} = (price_high \times 1.5 / 3600) \times vm.c.(n + i) / 2 )</td>
</tr>
<tr>
<td>5 ( vm.n_{(n+i)} = i )</td>
</tr>
<tr>
<td>6 ( vm.t_{(n+i)} = ct.l_i / (vm.s_{(n+i)} \times 2.5 \times 1000 \times 0.9) )</td>
</tr>
<tr>
<td>7 ( vm.e_{(n+i)} = 1 )</td>
</tr>
</tbody>
</table>
5 Test Results

In this section, CloudSim is used to implement the HRFDC strategy. At the same time, the IRW algorithm\cite{26}, RI algorithm\cite{27} and DTRDT algorithm\cite{28} are simulated and implemented. Later, a comparison module was added to the platform, which can compare the failure repair rate of CDCs, compensation of cloud service providers and the profit of cloud service providers between different mapping algorithms. At the same time, the method generates corresponding CT parameters for simulation \cite{29–31}. Other VM parameters refer to the real VM parameters provided by Amazon. As shown in Table 8. The failure repair rate of CDCs, the compensation of cloud service providers and the profit of cloud service providers will be verified in this section. The failure repair rate of CDCs is the proportion of all FCTs that are completed on time after rescheduling. The compensation of cloud service providers is the cost of the VM created for the execution of the FCT and the compensation for breach of contract. The profit of cloud service providers is the cost of the CT minus the compensation of the cloud service provider. The failure repair rate of CDCs is as follows:

\[
\text{Fault\_repair} = \frac{n_1}{n_2}
\]

4

Among them, SLA\_deadline represents the failure repair rate of CDCs. The \( n_1 \) represents the number of repaired faults. The \( n_2 \) represents the number of all faults.

The profit of cloud service providers are as follows;

\[
\text{profit} = \sum_{j=1}^{m} (ct\_p\_j - ct\_f\_j)
\]

5

Among them, profit represents the profit of cloud service providers. The \( ct\_p\_j \) represents the profit of the CT numbered \( j \) ($).
Table 8
experimental parameters

<table>
<thead>
<tr>
<th>parameters</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length of the CT(MI)</td>
<td>100000–1000000</td>
</tr>
<tr>
<td>Memory size of the CT(GB)</td>
<td>{8, 16, 32}</td>
</tr>
<tr>
<td>Number of the CT</td>
<td>{10000, 20000, 30000, 40000, 50000}</td>
</tr>
<tr>
<td>Failure rate of VM</td>
<td>{0.01, 0.05, 0.10}</td>
</tr>
<tr>
<td>Price of VM($/h) (low_high, price_high)</td>
<td>{(0.086, 0.119), (0.108, 0.146), (0.112, 0.153), (0.096, 0.133)}</td>
</tr>
<tr>
<td>Maximum memory of VM(GB)</td>
<td>{64, 96, 192, 384}</td>
</tr>
</tbody>
</table>

5.1 Different numbers of CTs

Failure rate of VM is 0.01. Price of VM is (0.086, 0.119). Maximum memory of VM is 64. Other parameters are shown in Table 8.

As can be seen from Fig. 2 (A), the failure repair rate of the HRFDC strategy is close to 100%. This is mainly because the VM that meets the deadline can be searched in order from small to large according to the dynamic classification rule of FCT when selecting VMs. At the same time, increasing the frequency and increasing the ratio of CPU can speed up the execution speed of CTs. The IRW algorithm still uses the original configuration when selecting VMs. Therefore, a large number of FCTs exceed the deadline. The RI algorithm adds copy replication technology. It adds 10% copies of VMs in the CDC. Although this can improve the repair rate of some faults, but the effect is not obvious. The DTRDT algorithm simply expands the VM configuration when selecting VMs. It does not use the method of increasing the frequency and ratio of CPU so that the fault repair rate is slightly inferior to the HRFDC strategy. The effect of the four algorithms is more stable as the number of CTs increases. It can be seen from Fig. 2 (B) that cloud service providers of the RI algorithm has the largest compensation, mainly because it uses the copy replication technology to add additional redundancy. The compensation of cloud service providers of IRW algorithm and DTRDT algorithm is slightly higher than that of HRFDC strategy. Mainly because of their low failure repair rate, resulting in a large amount of compensation. It can be seen from Fig. 2 (C) that the profit of cloud service providers is exactly the opposite of the compensation of cloud service providers. The HRFDC strategy makes it the most profitable because of the gradual increase in the processing speed of the VM.

5.2 Different failure rate of VM

Number of the CT is 10000. Price of VM is (0.086, 0.119). Maximum memory of VM is 64. Other parameters are shown in Table 8.
It can be seen from Fig. 3 that the failure repair rate of CDCs, compensation of cloud service providers, and profit of cloud service providers are similar to Fig. 2 of HRFDC strategy. The reason has been described above, and will not be repeated here. However, with the increase in the failure rate of the proposed machine, compensation of cloud service providers and profit of cloud service providers of HRFDC strategy have been moderated. This is mainly because it uses a gradual increase in the processing speed of the VM.

5.3 Different price of VM

Number of the CT is 10000. Failure rate of VM is 0.01. Maximum memory of VM is 64. Other parameters are shown in Table 8.

Figure 4 selects the real price of Amazon's four CDCs in the US East (Ohio), Asia Pacific (Singapore), Asia Pacific (Tokyo) and Europe (Ireland). We can see that no matter which DC, the failure repair rate of HRFDC strategy is higher than the other three algorithms. The compensation of the corresponding cloud service providers is also smaller than the other three algorithms. The profit of the cloud service providers is also greater than the other three algorithms. This also illustrates the versatility of the strategy proposed in this article.

5.4 Different maximum memory of VM

Number of the CT is 10000. Failure rate of VM is 0.01. Price of VM is (0.086, 0.119). Other parameters are shown in Table 8.

As can be seen from Fig. 5, the failure repair rate of the HRFDC strategy and DTRDT algorithm will increase with the increase of the maximum memory of VM. This is mainly because the configuration of the VM needs to be expanded in the same proportion in two algorithms. As the maximum memory of VM increases, there are more large-memory VMs to choose, which can further repair some problems with larger memory or shorter deadlines. The other change trends are the same as before and will not be repeated here.

6 Conclusions And Future Work

We first analyzed the overall architecture and model of the CDCs in this article. Secondly, the initial mapping of CTs and VM failures are studied. Thirdly, the HRFDC strategy based on dynamic classification of FCTs is proposed. Finally, the real data of Amazon CDC is used to verify the effect of the HRFDC strategy proposed in this article. It can be seen from the comparison that compared with the IRW, RI and DTRDT algorithms, the HTVM2 strategy greatly improves the fault repair rate of the CDCs and reduces the compensation of cloud service providers. Eventually, the profit of cloud service providers is improved.

There are two main directions for future work. First, we combine the parameters of multi-CDCs and the cost of electricity in different regions to further study the rescheduling problem of multiple DC. Second,
we study the rescheduling problem of multi-user and cross-geographical areas to weigh the balance between energy consumption and revenue.

**Declarations**

**Ethical Approval**

Not applicable

**Competing interests**

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

**Authors' contributions**

Bin Liang: Conceptualization, Methodology, Software, Validation, Writing - Original Draft Junqing Bai: Visualization, Supervision, Formal analysis, Writing - Review & Editing

**Funding**

This work was supported by the Key R & D Plan of Shaanxi Province (General Project) [No. 2022GY-031] and the Science and Technology Program of Xi’an [No. 2020KJRC0101].

**Availability of data and materials**

Not applicable

**References**


**Figures**

![m5a.large(Universal)](image)

**Figure 1**
result after CT mapping
Figure 2

Experimental results on different numbers of CTs
Figure 3

Experimental results on different failure rate of VM
(A) The failure repair rate of CDCs on different price of VM

(B) The compensation of cloud service providers on different price of VM

(C) The profit of cloud service providers on different price of VM

Figure 4

Experimental results on different price of VM
Figure 5

Experimental results on different maximum memory of VM

(A) The failure repair rate of CDCs on different maximum memory of VM

(B) The compensation of cloud service providers on different maximum memory of VM

(C) The profit of cloud service providers on different maximum memory of VM