Article

A Task Orchestration Strategy in a Cloud-Edge Environment Based on Intuitionistic Fuzzy Sets

Chunmei Huang [1], Bingbing Fan [1,*] and Chunmao Jiang [2]

[1] School of Computer Science and Information Engineering, Harbin Normal University, Harbin 150025, China; huangchunmeihl@163.com
[2] School of Computer Science and Mathematics, Fujian University of Technology, Fuzhou 350118, China; jiang@fjut.edu.cn
* Correspondence: fbbhsd@163.com

Abstract: In the context of the burgeoning cloud-edge collaboration paradigm, powered by advancements in the Internet of Things (IoT), cloud computing, and 5G technology, this paper proposes a task orchestrating strategy for cloud-edge collaborative environments based on intuitionistic fuzzy sets. The proposed strategy prioritizes efficient resource utilization, minimizes task failures, and reduces service time. First, WAN bandwidth, edge server virtual machine utilization, delay sensitivity of the task, and task length are used to determine whether the task should be executed on the cloud or edge device. Then, the cloud-edge collaborative decision-making algorithm is used to select the task's target edge servers (either the local edge servers or the neighboring edge servers). Finally, simulation experiments are conducted to demonstrate the effectiveness and efficacy of the proposed algorithm.

Keywords: edge computing; task orchestration; intuitionistic fuzzy sets; fuzzy inference; membership function; non-membership function

MSC: 94D05

check for updates

Citation: Huang, C.; Fan, B.; Jiang, C. A Task Orchestration Strategy in a Cloud-Edge Environment Based on Intuitionistic Fuzzy Sets. *Mathematics* **2024**, *12*, 122. https://doi.org/10.3390/math12010122

Academic Editor: Xibei Yang

Received: 20 November 2023
Revised: 26 December 2023
Accepted: 26 December 2023
Published: 29 December 2023

1. Introduction

One of the hot areas for research and application right now is edge computing, an emerging computing paradigm caused by the rapid advancement of IoT, cloud computing [1], 5G, and other technologies. To reduce network latency and improve application performance and security, edge computing shifts processing and storage resources to the edge of the data [2]. At the moment, extensive research and applications on edge computing are being conducted across a variety of sectors and domains. Researchers in academia are exploring the ideas and tenets of edge computing and its practical applications. They are working on new algorithms and models to perform complex computing tasks on edge devices and improving edge devices' performance and reliability. Large technological corporations are currently investigating edge computing applications in the sector. To facilitate the development and implementation of various applications, including industrial IoT, smart cities, and self-driving cars, they are creating and implementing a range of edge devices and platforms [3].

Meanwhile, governments and standardization organizations are promoting the development and adoption of edge computing. Standards organizations are creating a set of guidelines and standards to guarantee the security and interoperability of edge computing systems, while government agencies are creating policies to encourage edge computing research and use. Edge computing research and applications are developing, and in the future, they will become more significant across a range of sectors and domains. As an essential node of edge computing, the edge device performs more and more tasks such as computation, storage, communication, and security, and the requirements for its performance and quality are becoming greater and greater. So, cloud-edge collaboration is now introduced to reduce pressure on the edge node. The cloud will share a portion of the work execution, but how to effectively schedule and organize these tasks is a crucial concern to guarantee the performance and quality of computing.

The following issues primarily affect task orchestration for cloud-edge collaboration. The first is that edge devices are frequently uncertain and dynamic. For example, their capabilities, resource usage, network environment, and other factors can change over time. The second is that the edge tasks are complex and varied. Their types, sizes, and other characteristics vary, necessitating flexible orchestration in the actual situation. The orchestration of tasks in cloud-edge collaboration faces key challenges due to the dynamic nature of edge devices and the complexity of edge tasks. This dynamic environment requires a flexible approach to manage the varying types, sizes, and characteristics of tasks.

Recent approaches have focused on better decision making using three-way decisions [4–7], rough set theory [8–11], and intuitionistic fuzzy sets to handle uncertain information. These methods effectively deal with uncertain and incomplete data. Ignoring neutral uncertainties during decision making can result in insufficient information, leading to suboptimal decision results. Intuitionistic fuzzy sets precisely compensate for this drawback by using hesitation values. Thus, in the face of inaccurate information or insufficient data, an intuitionistic fuzzy set is a valuable mathematical tool.

Therefore, to achieve efficient task scheduling in cloud-edge environments, this paper introduces intuitionistic fuzzy set description variables and makes decisions on task scheduling based on the fuzzy reasoning results of intuitionistic fuzzy sets.

The remainder of this paper is organized as follows. Section 2 reviews current research on task orchestration in a cloud-edge environment, the definition of intuitionistic fuzzy sets, and cloud-edge architecture. Section 3 introduces a task orchestrating algorithm grounded in intuitionistic fuzzy sets in a cloud-edge environment. Finally, Section 4 presents experimental results and analyses, including performance evaluations and practical experiments, to demonstrate the efficacy of our proposed algorithm.

2. Related Work

This section will introduce the background of task scheduling in a cloud-edge environment, the relevant intuitionistic fuzzy set theory, and the cloud-edge architecture adopted.

2.1. Task Orchestrating in a Cloud-Edge Collaboration Environment

An essential topic of study in edge computing is task orchestration for cloud-edge collaboration. Task orchestration seeks to balance workloads among nodes and assign tasks to the most effective computing nodes to maximize system performance and resource efficiency. Researchers have developed many offloading models and optimization techniques to address the issue of task offloading in edge environments in recent years. These techniques are currently mainly divided into those that consider only the edge layer and those that consider only the cloud layer. According to specific optimization objectives, they can be roughly divided into the following three [12]: minimizing energy consumption, minimizing delay, and improving service quality.

The first category of algorithms is designed to minimize energy consumption, which is crucial in cloud-edge computing. Fan et al. [13] and Chen et al. [14] have made significant strides in this area. Fan et al. developed an optimized hierarchical cloud-FEC network, while Chen et al. focused on a genetic algorithm for energy-efficient offloading. However, these solutions primarily focus on energy savings, often not addressing the broader impacts on network and computing resource efficiency. This narrow focus can be a limitation in improving overall service quality in cloud-edge environments.

Another pursuit of efficient task orchestration in cloud-edge collaboration faces the challenge of latency reduction and quality optimization in dynamic environments. Dou et al. [15] addressed latency through a mixed integer nonlinear optimization problem, focusing on task offloading and resource allocation. Niu et al. [16] developed the

BRT algorithm to minimize service delay. However, these algorithms often target a single optimization goal, overlooking the dynamic and uncertain nature of practical applications.

To address dynamic task scheduling environments, the third category approach involves optimizing overall service quality. Sonmon et al. [17] utilized fuzzy logic for two-stage decision making in edge computing, adapting to changing conditions. Ali et al. [18] applied a fuzzy logic-based task orchestrating algorithm, considering task requirements and constraints. However, these methods may not fully capture the hesitancy and uncertainty in expert evaluations, potentially leading to suboptimal decisions.

This paper designs a task orchestrating strategy for cloud-edge collaboration based on intuitionistic fuzzy sets, which can more efficiently deal with the problem of hesitancy in the actual scenarios based on the consideration of the dynamically changing environments. The intuitionistic fuzzy set is an extension of fuzzy sets consisting of intuitionistic fuzzy numbers. Intuitionistic fuzzy sets are composed of membership degrees, non-membership degrees, and hesitancy degrees, which are more flexible than fuzzy sets in representing uncertainties. When it is unsure whether the value belongs to a membership degree or a non-membership degree, it is considered a hesitancy degree, giving more scope to represent the uncertainty. The proposed algorithm solves problems such as the single optimization objective of task orchestrating in the cloud-edge environment and the inability to better deal with uncertainties.

2.2. Intuitionistic Fuzzy Set Theory

In recent years, there have been many applications of fuzzy theory. Ciric et al. [19] proposed a formal decision-making process based on intuitionistic fuzzy theory to achieve mechanized selection and rational use. Besiktepe et al. [20] have developed a resource-efficient quantitative CA framework based on fuzzy theory, which can reduce subjectivity when establishing conditional ratings. Wozniak et al. [21] proposed an intelligent control system based on fuzzy logic type-two for humidity control and regulation. They apply type-two fuzzy logic to system inference and decision making to adapt to complex and uncertain humidity environments.

An expansion of fuzzy set theory, intuitionistic fuzzy sets are designed to address specific issues that arise in the real-world applications of fuzzy set theory. Intuitionistic fuzzy sets are currently the subject of extensive study that spans numerous domains and use cases. Theoretically, many studies have been conducted on intuitionistic fuzzy sets' notions, properties, and inference methods. Numerous varieties of intuitionistic fuzzy sets, including binary and ternary versions, have been proposed by researchers. Numerous applications for intuitionistic fuzzy sets include data mining, intelligent control, pattern detection, and many more domains. For instance, researchers have segmented and recognized image regions using intuitionistic fuzzy sets in the field of pattern recognition, with improved results [22]; in the field of intelligent control, intuitionistic fuzzy sets have been applied to controller design, which can improve controller performance and stability [23]; in the field of data mining, intuitionistic fuzzy sets can be used to process multiattribute data, classify and cluster data, and so forth [24]. In conclusion, intuitionistic fuzzy sets, as an extension of fuzzy set theory, have been widely studied and applied in theory and application. The traditional fuzzy set is defined as [25]:

$$A = \left\{ < x,\ \mu_{A(x)} > \mid\ x \in\ X \right\}, \tag{1}$$

where $\mu_{A(x)}$ is the degree of membership, the degree to which the element belongs to the set, and these degrees of membership can be any real number between 0 and 1. Extending the concept of classical sets to treat the degree of membership of an element as a continuous quantity leads to a better representation of fuzzy and uncertain concepts in the real world. The concept of intuitionistic fuzzy sets is defined as follows [26]:

$$A = \left\{ < x,\ \mu_{A(x)},\ v_{A(x)} > \mid\ x \in X \right\}, \tag{2}$$

where $\mu_{A(x)}, v_{A(x)} \in [0,1]$, and satisfies $0 \leq \mu_{A(X)} + v_{A(x)} \leq 1, \forall\, x \in X$, and $\mu_{A(x)}$ and $v_{A(x)}$ are the degree of membership and non-membership, respectively, where $\mu_{A(x)} :\ x \to [0,1], v_{A(x)} :\ x \to [0,1],\ x \in X, \pi_{A(x)} = 1 - \mu_{A(x)} - v_{A(x)}$ is called the hesitation degree and also satisfies $0 \leq \pi_{A(x)} \leq 1$.

For example, there is an intuitionistic fuzzy set $A = \{< x, 0.2, 0.6 > \mid\ x \in X\}$, which means that x belongs to A with degree 0.2, x does not belong to A with degree 0.6, and is neutral with degree 0.2. As an example to understand intuitionistic fuzzy sets through simple examples in daily life, there are three types of voting: support, opposition, and neutrality. Under certain conditions, neutral individuals can become definite supporters or opponents. Therefore, considering neutral individuals can better describe practical problems. It corresponds to the membership degree, the non-membership degree, and hesitation degree, respectively. Therefore, if we want to characterize an intuitionistic fuzzy set, we need to use at least any two of the functions of the membership function, the non-membership function, and the degree of hesitancy.

2.3. *Cloud-Edge Architecture*

In the field of cloud-edge architecture, initial research focused primarily on task scheduling within cloud layers. However, the increasing volume of tasks led to network congestion and slower cloud processing, resulting in higher task failure rates and prolonged processing times. This situation necessitated the emergence of an edge layer, designed to alleviate the load on cloud resources by handling data transmission and processing more efficiently.

The cloud-edge architecture described in this paper adopts a multi-tier approach [27]. The model, illustrated in Figure 1, features a cloud center layer at the top, a middle edge layer, and a device layer at the bottom composed of various mobile devices. Within this structure, the edge layer, despite its limited capabilities, plays a crucial role in reducing the burden on the cloud layer. However, overloading the edge layer with tasks can also lead to increased failure rates, highlighting the need for a balanced approach in task allocation.

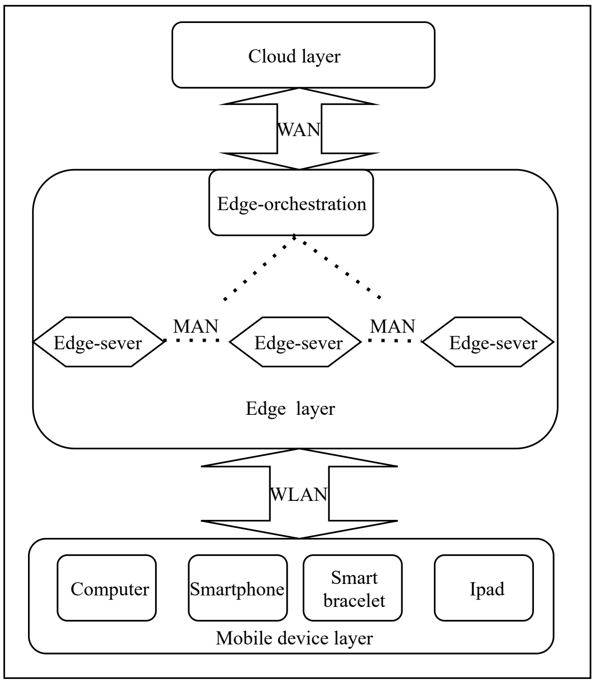


Figure 1. Cloud-edge structure.

Each layer of the architecture has a distinct function. The mobile device layer, including smartphones and tablets, handles simple, real-time tasks such as sensor data analysis, but is limited in computational and storage capabilities. The edge layer, comprising several dispersed nodes with computing and communication abilities, processes and stores real-time, low-latency data. The edge orchestrator layer, a programmable and distributed platform, dynamically manages computational and service resources across the edge network to ensure high performance and availability. Finally, the cloud layer, which serves as a hub for cloud computing, management, and storage, tackles complex tasks such as big data analysis and deep learning, with outcomes relayed back to the edge nodes or end devices.

This cloud-edge collaborative architecture is well-suited to meet the demands of the Internet of Things, the mobile Internet, and similar applications. It offers flexible resource management, low-latency responses, and efficient data processing, supporting a range of emerging industry applications such as smart manufacturing, smart cities, and intelligent transportation. Communication within this architecture is facilitated by a wireless local area network (WLAN) between the mobile device layer and the edge layer, a metropolitan area network (MAN) connecting edge devices, and a wide area network (WAN) for interactions between devices in the edge layer.

2.4. Problem Statement

Each user node in the network generates a set of tasks, denoted as $T = \{t_1, t_2, t_3, \ldots, t_i\}$. The edge orchestrator determines the execution location for each task. The decision-making process in both stages is implemented by intuitionistic fuzzy reasoning. The result of fuzzy reasoning is defined as IFS, and thresholds d_1 and d_2 are set. This decision making is formalized by two functions:

Cloud or Edge Layer Decision:

$$f1 : T \to \{\text{Cloud}, \text{Edge}\}. \tag{3}$$

$$where\ \ f1(t_i) = \begin{cases} \text{Cloud} & \text{if } IFS(t_i) > d_1, \\ \text{Edge} & \text{otherwise.} \end{cases} \tag{4}$$

Local or Neighboring Edge Server Decision:

$$f2 : T \to \{\text{Local Edge Server}, \text{Neighboring Edge Server}\}. \tag{5}$$

$$where\ \ f2(t_i) = \begin{cases} \text{Neighboring Edge Server} & \text{if } IFS(t_i) > d_2, \\ \text{Local Edge Server} & \text{otherwise.} \end{cases} \tag{6}$$

3. Collaborative Task Orchestrating Algorithm for Cloud-Edges Based on Intuitionistic Fuzzy Theory

This paper combines intuitionistic fuzzy theory with task scheduling in a cloud-edge environment. The general process of intuitionistic fuzzy reasoning is introduced in this section, and task scheduling algorithms in a cloud-edge collaborative environment are proposed.

3.1. Intuitionistic Fuzzy Reasoning

The intuitionistic fuzzy inference [28] employed in this work is the outcome of linearly combining the outputs of the two fuzzy systems after performing fuzzy inference on the membership and non-membership functions independently. One can compute the final output as follows:

$$IFS = (1 - \pi) * FS_\mu + \pi * FS_v, \tag{7}$$

where FS_μ is the result of fuzzy inference of the membership function, FS_v is the result of non-membership function fuzzy reasoning, and π is the degree of hesitation in the intuitionistic fuzzy set. It is clear that intuitionistic fuzzy reasoning degrades to traditional fuzzy reasoning when $\pi = 0$. Intuitionistic fuzzy inference is divided into four steps of the following specific process:

Step 1: Intuitionistic Fuzzification

Establish the membership and non-membership function of the input and output variables using the trapezoidal and the triangular membership function. For each input variable, an intuitionistic fuzzy set $A^* = \{< x, \mu_{A^*(x)}, v_{A^*(x)} > | x \in X\}$ is established, in which $\mu_{A^*(x)}$ and $\pi_{A^*(x)}$ is generally given by the expert experience, then $v_{A^*(x)} = 1 - \mu_{A^*(x)} - \pi_{A^*(x)}$.

Step 2: Intuitionistic Fuzzy Reasoning

For the established intuitionistic fuzzy set A^* and intuitionistic fuzzy relationship (IF-THEN rule), the Mamdani fuzzy inference system is used to infer the membership and non-membership functions of the input variables, respectively.

Step 3: Defuzzification

Applying the center of gravity method of defuzzification for the reasoning results in the above steps, the value is given by:

$$w = \frac{\int x\mu(x)dx}{\int \mu(x)dx}. \tag{8}$$

Step 4: Output linear results

The final output of the intuitionistic fuzzy inference system, IFS, is calculated according to Equation (7).

Two algorithms are designed in this paper using intuitionistic fuzzy reasoning; the first uses the edge layer orchestrating algorithm to obtain the task's target edge servers (either the local edge servers or the neighboring edge servers), and the second uses the cloud-edge collaborative decision-making algorithm to determine whether the task should be executed on the cloud or edge device based on the WAN bandwidth, the edge server VM utilization, the task latency sensitivity, and the task length.

3.2. Task Orchestrating Based on Intuitionistic Fuzzy Reasoning in Cloud-Edge Environments

In the edge computing environment, the decision of where to unload tasks is affected by many factors, such as edge device utilization, network environment, and computing resources. However, these influencing factors usually change dynamically in the uncertain edge computing environment. Uncertainty cannot be accurately expressed in mathematical terms. On the contrary, it can be described in a fuzzy and uncertain range. Intuitionistic fuzzy sets are easy to understand and apply, and the ideas and processes of intuitionistic fuzzy logic in this paper are referenced from [28]. It uses membership and non-membership degrees in fuzzy sets to describe the fuzziness and uncertainty of problems, which is more in line with the reality of the world and human thinking processes. Therefore, intuitionistic fuzzy logic can generate decision results that are easier to understand and explain, enabling the system to interact and communicate with people effectively.

3.2.1. Input Variables

Numerous factors affect the decision-making phase. We use the following factors as input variables [17] for the intuitionistic fuzzy reasoning phase: WAN bandwidth, edge device VM utilization, delay sensitivity of the task, and task length. It can be stated as:

$$F = (W, U, S, L). \tag{9}$$

In this case, W stands for WAN bandwidth, U for edge server virtual machine utilization, S for task latency sensitivity, and L for task length. The four parameters affecting the task orchestrating process are considered while determining the target server for a task.

In the design of this article, the WAN bandwidth W significantly impacts the transmission with the cloud layer. One way to implement task offloading is to transfer a portion of the computational workload to the cloud. WAN bandwidth directly impacts task offloading efficacy since it necessitates data transmission over WAN for task offloading. Offloading tasks from the edge device to the cloud will increase transmission latency, slow offloading, and perhaps result in unsuccessful offloading if the WAN bandwidth falls below a specific threshold.

U is the utilization of edge server virtual machines in edge computing. U determines whether tasks should be delegated to the cloud or the edge. It may be possible to fully utilize the edge server's computational resources by offloading tasks if its utilization is low. By offloading work to the cloud, we can fully leverage the cloud's computing capabilities and balance the load and performance of the entire edge computing system, especially if the edge servers are being used to a high degree.

The term "delay sensitivity S" describes a task's sensitivity to its processing speed within a relatively constrained processing period. This is particularly true for jobs that are based on real-time input from sensors, like autonomous driving, where information about the state of the road is evaluated and responded to promptly to guarantee safety. Edge devices are better suited for tasks that are more responsive to latency because they can process and respond to commands more quickly.

Offloading tasks to edge nodes can considerably reduce latency because the data may be processed closer to the user when tasks are short. The task length, L, and execution time are strongly connected. On the other hand, cloud offloading can be more suitable for lengthy jobs. The cloud's tremendous computation and storage capabilities can easily solve large-scale computing processes. Long tasks usually need additional processing power and storage capacity, and the cloud can supply more processing capacity to satisfy these demands.

3.2.2. Step 1: Intuitionistic Fuzzification

The step of transforming the input variables' unambiguous values into fuzzy values is known as intuitionistic fuzzification [29]. The Mamdani fuzzy inference system [30] is utilized in this work because of its simplicity and broader applicability. Triangular and trapezoidal membership functions are used to determine each input variable's membership values and levels, first establishing the membership and non-membership functions of the input variables [31]. Next, the membership function image is used to blur the clear values of the input variables, resulting in fuzzy variables. The appropriate language variables for the input variables should be defined. This method's linguistic variables correspond to low, medium, and high for the four input variables: WAN bandwidth, edge device VM utilization, task delay sensitivity, and task length. The values of the membership function are generally obtained from the experience of experts, and the values of the membership function of the input variables used in this paper synthesize the values of the experts' experience from [17] as well as from [27]. To simplify the calculation, the degree of hesitancy π is set to 0.1, and the non-membership function can be calculated as $v_{A(x)} = 1 - \mu_{A(x)} - \pi_{A(x)}$. Figure 2 displays the membership function images for the input variables WAN bandwidth and task length, whereas Figure 3 illustrates the non-membership function images.

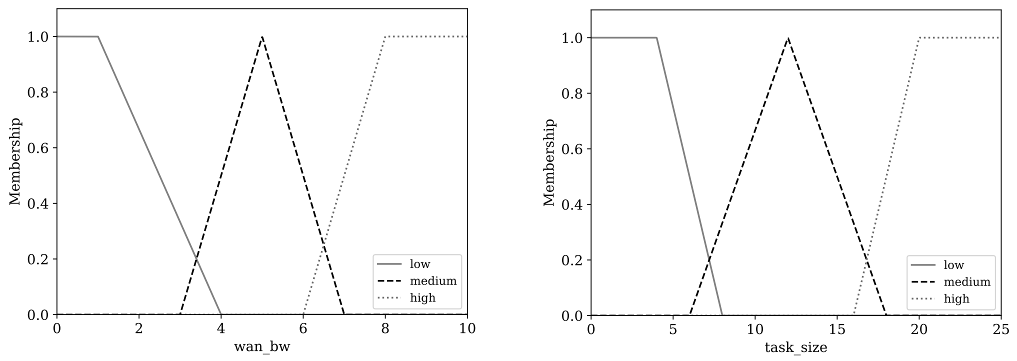


Figure 2. The membership functions of WAN bandwidth and task size.

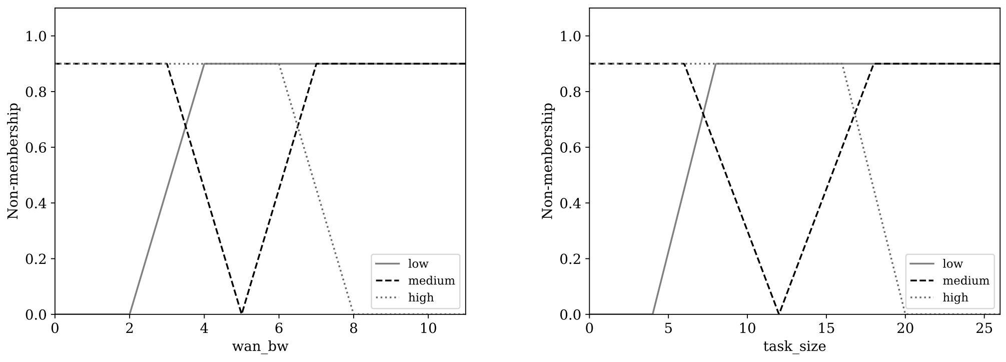


Figure 3. The non-membership functions of WAN bandwidth and task size.

3.2.3. Step 2: Intuitionistic Fuzzy Reasoning

Fuzzy variables in the reasoning machine are calculated using established fuzzy rules to produce a new fuzzy variable. A fuzzy rule base consists of a set of fuzzy rules similar to human reasoning processes. It is a simple if-then rule that covers all possible scenarios of application features and system conditions. These rules play a crucial role in defining overall system performance. An example of a rule is to offload a task to the cloud if it has a low edge VM utilization, low task length, medium WAN bandwidth, and low delay sensitivity. The output value will be used for the decision stage. Table 1 provides a part of the system's fuzzy rules. The main goal is to avoid node overload and provide low latency for edge applications, affecting service time and task failure rate.

Table 1. Partial fuzzy rules.

Rules	Edge VM Utilization	Length of Task	WAN-bw	Delay Sensitivity	Decision
1	low	low	low	low	edge
2	low	low	low	medium	edge
3	low	low	low	high	edge
4	low	low	medium	low	cloud
5	low	low	medium	medium	edge
6	low	low	medium	high	edge

A fuzzy logic if-then rule base with 81 rules is developed in our decision-making algorithm with four input and two output variables. In this step, three stages of aggregation, activation, and accumulation are necessary to obtain the new fuzzy value in the inference machine.

3.2.4. Step 3: Defuzzification

Defuzzification is the third stage. The process of turning the fuzzy output that the inference machine inferred into crisp values is called defuzzification. Numerous other defuzzification techniques exist, and the center of gravity method is employed in the proposed algorithm. FS_{μ} and FS_{v} can be computed using Equation (8).

3.2.5. Step 4: Output Linear Results

In this thesis, the threshold value d used to determine the location of task offloading is set to 50. When the IFS value exceeds 50, the task will be offloaded to the cloud server; otherwise, the edge orchestrator will schedule the task offloaded to the appropriate edge server. This brings us to the decision-making phase. The decision value of the decision-making phase can be obtained using Equation (7). Algorithm 1 illustrates the precise steps of the algorithm, while Figure 4 illustrates the algorithm flow.

Algorithm 1 : Cloud-edge task orchestration algorithm.

```
Input:  The real-time values of these variables
            $W$: WAN bandwidth
            $U$: Edge server VM utilization
            $S$: Latency sensitivity of the task
            $L$: Task length
            $d_1$: Threshold
Output:  IFS : Intuitionistic fuzzy reasoning result
            Optimal position for task orchestrating
1: Read Wan-bw, Avg-edge-util, Delay-sensitivity, Task-size
2: Fuzzy inference is performed on the membership and non-menbership function of the
   input variables to obtain $FS_\mu$ and $FS_V$
3: Calculate $IFS = (1 - \pi) * FS_\mu + \pi * FS_v$
4: If $IFS < d_1$ then
5:     Task offloading to edge server
6: Else
7:     Task offloading to cloud server
8: End
9: Return  Outputs
```

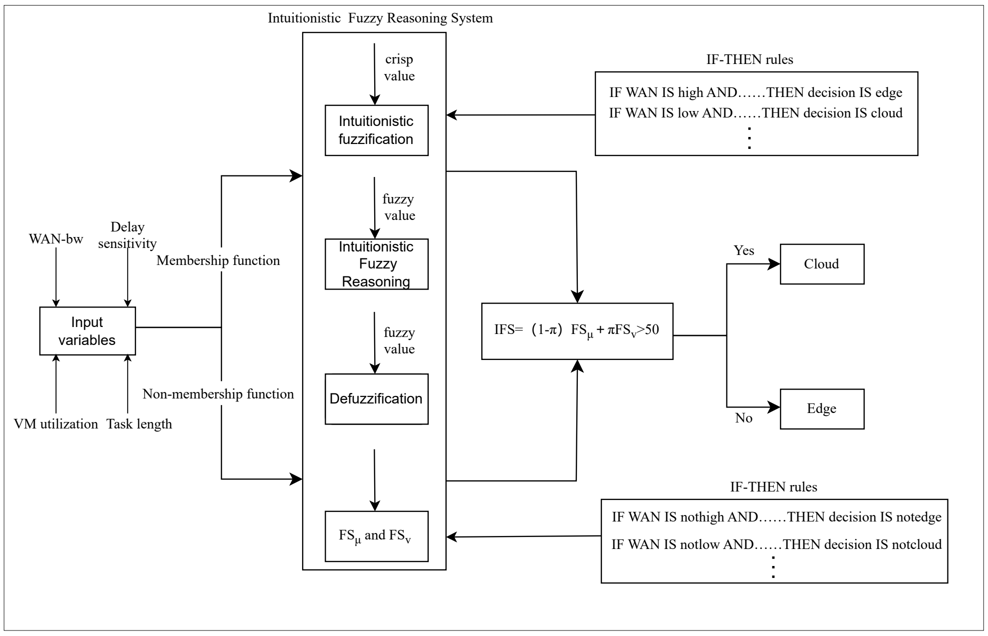

Intuitionistic Fuzzy Reasoning System
crisp value
Intuitionistic fuzzification
fuzzy value
Intuitionistic Fuzzy Reasoning
fuzzy value
Defuzzification
FS_μ and FS_v
IF-THEN rules
IF WAN IS high AND……THEN decision IS edge
IF WAN IS low AND……THEN decision IS cloud
WAN-bw
Delay sensitivity
Membership function
Input variables
Non-membership function
VM utilization
Task length
$IFS= (1-\pi) FS_\mu + \pi FS_v > 50$
Yes
Cloud
No
Edge
IF-THEN rules
IF WAN IS nothigh AND……THEN decision IS notedge
IF WAN IS notlow AND……THEN decision IS notcloud

Figure 4. Flow of task orchestrating based on intuitionistic fuzzy reasoning in cloud-edge environments.

3.3. *Edge Task Orchestrating Based on Intuitionistic Fuzzy Reasoning*

The edge orchestrator can choose whether to schedule a task to the cloud or the edge using Algorithm 1. This paper designs Algorithm 2, which determines whether to schedule a task to a local edge server or a neighboring edge server in the edge layer to fully utilize the edge server resources. The optimal edge server is chosen for the task in the edge layer by applying intuitionistic fuzzy reasoning to three input variables: the network delay, the local edge server's CPU utilization, and the neighboring edge server's CPU utilization. This helps to prevent some local edge servers from becoming overloaded during task execution while also allowing other neighboring edge servers in the edge layer with lower utilization to be taken into consideration in order to utilize the computational resources fully.

Algorithm 2 : Edge task orchestration algorithm.

```
Input:   The real-time values of these variables
           $N$: Network delay
           $C1$: CPU utilization of the local edge server
           $C2$: CPU utilization of neighboring edge server
           $d_2$: Threshold
Output:  IFS : Intuitionistic Fuzzy Reasoning Results
           Optimal position for task orchestrating
1: Read local edge util, neighboring edge util, network delay
2: Fuzzy inference is performed on the membership and non-menbership function of the
   input variables to obtain $FS_\mu$ and $FS_V$
3: Calculate $IFS = (1 - \pi) * FS_\mu + \pi * FS_v$
4: If $IFS < d_2$ then
5:     Task offloading to the local edge server
6: Else
7:     Task offloading to neighboring edge server
8: End
9: Return Outputs
```

3.3.1. Input Variables

The simulation scenario considers the following factors: the network delay, the local edge server's CPU utilization, and the neighboring edge server's CPU utilization. We employ low, medium, and high as linguistic variables for the three input parameters. This allows us to create a fuzzy rule base with 27 rules. Figure 5 displays the membership and non-membership function images of the input variable network delay.

In order to enhance computation efficiency and response time, edge computing must manage computation activities to be executed in local edge servers. This will minimize network latency and bandwidth usage between the computation tasks and the central servers. In order to make the most use of computational resources and minimize network congestion, task orchestrating may consider distributing computation tasks to neighbor computation nodes if the network latency is very low.

The utilization of nearby edge servers and local edge servers impacts task orchestrating for edge computing. In order to prevent overloading, make the best use of the server's resources, and increase computation speed and efficiency, the task orchestrating should think about distributing the computation tasks to other relatively idle edge servers if the local edge server's CPU utilization is high.

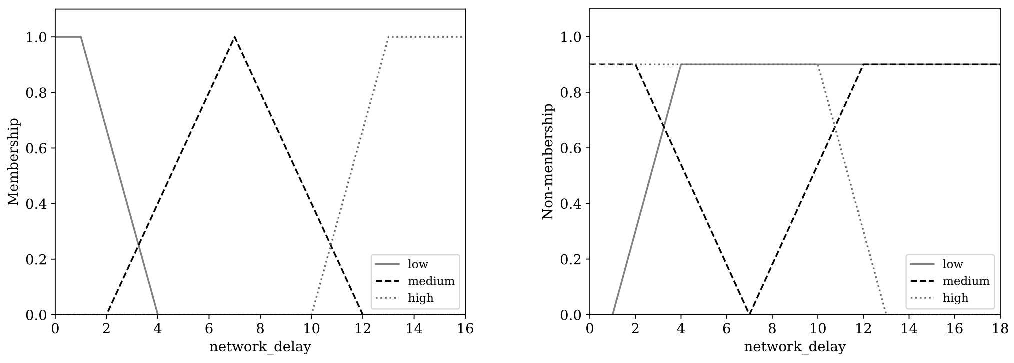


Figure 5. Membership and non-membership function of the network delay.

3.3.2. Edge Task Orchestration Algorithm

When the task is scheduled into the edge layer, it is determined whether it will be scheduled to the local edge servers or the neighboring edge servers based on the decision result of the output, which is compared with the size of the threshold after four steps of intuitionistic fuzzy reasoning, defuzzification, and calculating the linear output, respectively. Algorithm 2 displays the particular orchestrating algorithm.

4. Simulation Performance Evaluation

This section delves into a meticulous simulation study aimed at substantiating the efficacy of the proposed algorithm. The segment initially outlines the experimental setup and the evaluation metrics utilized, and subsequently juxtaposes the performance of our proposed algorithm against four other prominent algorithms in the domain.

4.1. Simulated Environment

For our simulation, we leveraged EdgecloudSim [32], an advanced Java-based simulation tool underpinned by the CloudSim [33] framework. EdgecloudSim is adept at modeling a layered architecture typical of edge computing, making it an ideal choice for scrutinizing various architectures' effects on task orchestration, resource allocation, and overall quality of service.

The simulations were carried out on a robust system featuring a 12th generation Intel(R) Core(TM) i5-12500H 2.50 GHz processor and 16 GB RAM, all running on a Windows 11 platform. We meticulously simulated four distinct types of application: enhanced, computational complexity, health, and entertainment applications. These were carefully selected to provide a holistic view of the algorithm's performance in a range of scenarios and are detailed in Table 2.

Furthermore, to position our algorithm within the context of the current edge computing landscape, we conducted a comparative analysis against several cutting-edge algorithms in the field. This comparison not only highlights the unique advantages of our proposed approach, but also provides a detailed performance benchmark, reaffirming its position as a potent solution for the evolving demands of cloud-edge computing environments.

Table 2. Parameters.

Parameter	Parameter Value
Time	33
Warm-up time	3
Virtual Machine Load Check Interval	0.1
Position check interval	0.1
Minimum number of devices	200
Maximum number of devices	2400
Number of hosts in cloud data centers	1
Number of virtual machines for cloud hosting	4
Number of cores of cloud VMs	4

4.2. Performance Metrics

In evaluating performance, the number of tasks is varied by altering the number of devices, and the outcomes are contrasted with several contemporary algorithms, including the fuzzy-competitor code offloading algorithm [34] and others. Our chosen metrics include the average task failure rate, service time, and average VM utilization, providing a comprehensive view of the algorithm's effectiveness.

Average Task Failure Rate (ATFR): $ATFR$ is defined as the ratio of the number of failed tasks to the total number of tasks. It is given by:

$$ATFR = \frac{\text{Number of Failed Tasks}}{\text{Total Number of Tasks}}. \tag{10}$$

Service Time (ST): Service time combines processing time and network latency. It is defined for a task i as:

$$ST_i = T_{\text{process},i} + T_{\text{latency},i}, \tag{11}$$

where $T_{\text{process},i}$ is the processing time and $T_{\text{latency},i}$ is the network latency for task i. The average service time over N tasks is then:

$$\overline{ST} = \frac{1}{N}\sum_{i=1}^{N} ST_i. \tag{12}$$

Average VM Utilization (AVMU): $AVMU$ is the average CPU usage of the virtual machines in the edge server, calculated as:

$$AVMU = \frac{1}{M}\sum_{j=1}^{M} \frac{\text{CPU Time Used by VM}_j}{\text{Total CPU Time Available for VM}_j}, \tag{13}$$

where M is the number of virtual machines, CPU Time Used by VM_j is the CPU time used by the j-th VM, and Total CPU Time Available for VM_j is the total available CPU time for the j-th VM.

4.3. Experimental Results and Performance Comparison

The experimental results of the proposed algorithm, as shown in Figure 6, demonstrate the great performance of the proposed algorithm in terms of service time and task failure rate, especially as the number of devices increases. This highlights the effectiveness of the algorithm in dynamic conditions and its potential for real-world applications.

In Figure 6a, all algorithms perform well when the number of task devices is small. However, as the number of devices increases, the proposed algorithm excels by considering various factors, such as network conditions and VM utilization, in relation to device overload. Dynamically selecting devices suitable for the current environment and arranging tasks more rationally ensures efficient resource utilization and improves the success rate of task completion.

Figure 6b compellingly demonstrates the superiority of the intuitionistic fuzzy logic-based algorithm, offering notably shorter service times compared to its counterparts. Service time, composed of both processing time and network latency, tends to be relatively short for several algorithms when device numbers are minimal. However, as device numbers swell and task volumes burgeon, our algorithm begins to show its true colors. It capitalizes on enhanced virtual machine utilization and effective task allocation, thus curtailing task completion delays and subsequently diminishing overall service time.

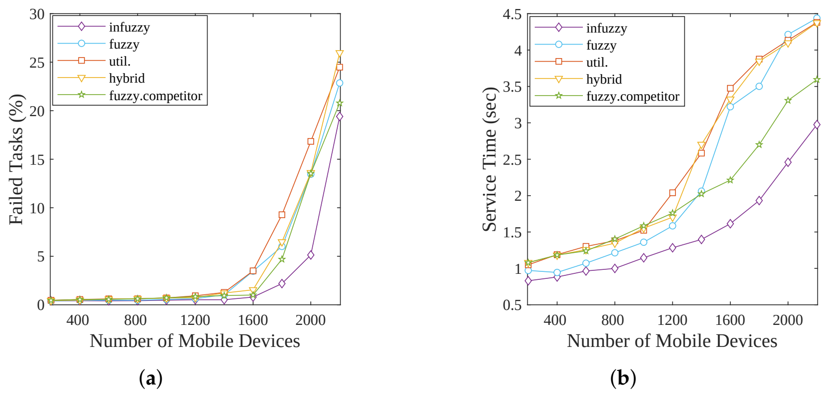
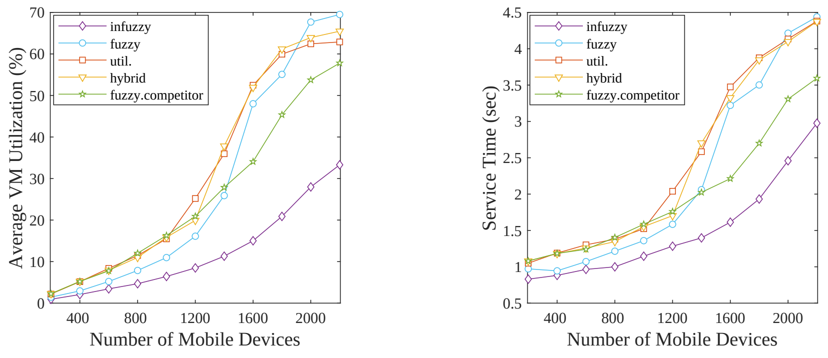


Figure 6. Comparative evaluation of the proposed algorithm based on all application types. (**a**) The failure rate; (**b**) The service time.

In the context of the average utilization of virtual machines depicted in Figure 7a, the performance of various algorithms appears marginal when the devices are few. In particular, our algorithm does not significantly outperform others in terms of average VM utilization under these conditions. However, as device numbers escalate, the edge CPU utilization under our intuitionistic fuzzy algorithm remains lower, suggesting optimal and full resource usage, thereby reflecting an uptick in overall resource efficiency. Figure 7b sheds light on the processing times of different algorithms on varying devices. When devices are few, processing times across different approaches are comparable. However, as device numbers proliferate, our algorithm maintains a more stable and efficient processing time. This stability is attributed to the superior handling of fuzzy input variables in dynamic computing environments by the intuitionistic fuzzy algorithm. It allocates computing resources more judiciously, ensuring that processing times remain consistently minimal, a testament to the algorithm's robust adaptability and efficiency in complex and dynamic cloud-edge ecosystems.



(**a**) (**b**)

Figure 7. Comparison of the proposed algorithm with others. (**a**) VM utilization; (**b**) The processing time.

The experimental study was benchmarked against four other sets of comparative analyses. The first set employed a fuzzy edge task arrangement that utilized conventional fuzzy methods for task scheduling. This approach primarily considered membership degrees while overlooking the hesitancy degree inherent in real-world expert decision making. As device numbers incrementally rose, our proposed strategy demonstrated increased virtual machine utilization, circumventing processing delays, and thereby reducing both service time and task failure rates more effectively. In the second comparative experiment, a bandwidth-based task scheduling strategy was assessed, focusing solely on the influence of bandwidth on the results. The third set involved a mixed task scheduling strategy, which integrated both bandwidth and edge server utilization considerations. However, because of their limited scope in accommodating dynamic factors, both the second and the third comparative experiments exhibited less robust results. The final comparative experiment drew on traditional fuzzy theory, considering variables such as edge server utilization, CPU speed, video execution rate, and task size. However, this approach did not comprehensively address task scheduling in a cloud-edge environment, leading to a less pronounced overall enhancement. Each of these comparative analyses underscores the superior adaptability and performance of our proposed intuitionistic fuzzy logic-based task scheduling strategy, particularly in dynamic and uncertain cloud-edge computing contexts.

In our study, we used the Edgecloudsim platform to simulate four different types of application tasks (APP), as depicted in Figure 8. Specifically, Figure 8a illustrates the service time of the Health App, which generates small tasks, while Figure 8b details the service time of the Augmented Reality (AR) App, known for producing medium-sized tasks with a low tolerance for latency. The results indicate that our proposed algorithm consistently maintains a lower percentage of service time, highlighting its effectiveness. The primary objective of our experiment was to reduce the task failure rate and enhance resource allocation efficiency by leveraging an intuitionistic fuzzy process. The data clearly show that as the number of devices increases, our method outperforms others by achieving the lowest service time across various types of tasks generated by different apps. This efficiency is attributed to the algorithm's ability to optimize virtual machine (VM) utilization, thereby reducing processing time and service time. Furthermore, Figure 8c,d elucidate the task failure rates for heavy computing applications and infotainment applications in different algorithms. Computing-intensive tasks, typically large-scale, require careful allocation to prevent device overload. Our approach effectively distributes tasks between cloud and edge devices even under such conditions, dynamically considering bandwidth and device utilization, significantly reducing processing latency, and consequently enhancing the task completion rate. This customized allocation strategy is a testament to the robustness and adaptability of our proposed algorithm, making it a viable solution for diverse and demanding cloud-edge computing environments.

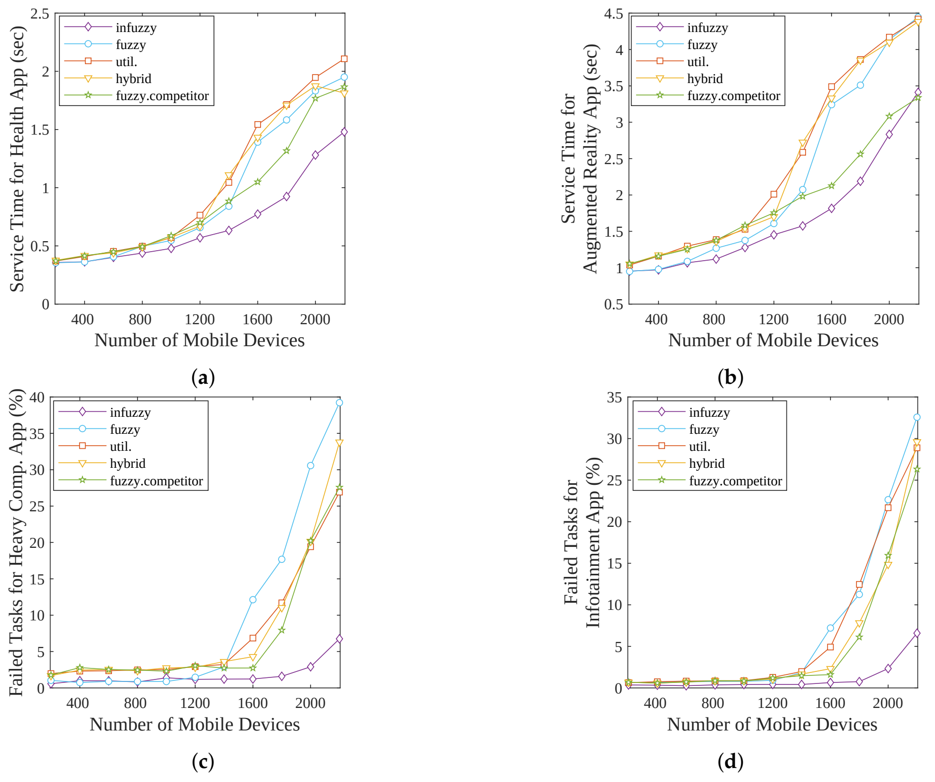


Figure 8. Comparative evaluation of the proposed mechanisms based on all application types. (**a**) Service Time for Health App (sec); (**b**) Service Time for Augmented Reality App (sec); (**c**) Failed Tasks for Heavy Comp. App; (**d**) Failed Tasks for Infotainment App.

5. Conclusions

This research delves into cloud-edge collaboration, introducing a task orchestrating approach grounded in intuitionistic fuzzy sets. Through nuanced consideration of WAN bandwidth, edge server virtual machine utilization, task latency sensitivity, and task length, the proposed method mirrors the real-world scenario more closely, where hesitation and ambiguity often exist. It effectively improves task execution efficiency and optimizes resource utilization within edge computing environments. Comparative analysis with alternative algorithms demonstrates that the approach presented here effectively manages uncertainty through intuitionistic fuzzy logic, leverages the cloud-edge architecture with an orchestrator for improved task resource management, and significantly boosts the task completion success rate. The viability and efficacy of this novel approach are further validated through comprehensive simulation experiments.

In the future, we will pave the way for further explorations and improvements. One notable limitation of the current study is the assumption of task independence, overlooking potential interdependencies among tasks. Future research will aim to address this gap, exploring the intricacies of task dependency in cloud-edge environments. This includes developing more sophisticated orchestration strategies that can dynamically adapt to the complex interplay of tasks, thereby further enhancing the decision-making process and overall system performance. Furthermore, expanding the scope to include real-world applications and more diverse computing scenarios will provide a more comprehensive understanding of the potential and limitations of the proposed approach.

Author Contributions: Conceptualization, B.F.; methodology, C.J.; validation, C.H.; formal analysis, C.H.; writing—original draft, B.F.; writing—review & editing, C.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Acknowledgments: This work was supported in part by Fujian University of Technology Research Fund Project (GY-Z220212).

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

References

1. Jiang, C.; Duan, Y.; Yao, J. Resource-utilization-aware task scheduling in cloud platform using three-way clustering. *J. Intell. Fuzzy Syst.* **2019**, *37*, 5297–5305. [CrossRef]
2. Rasheed, A.; Chong, P.H.J.; Ho, I.W.H.; Li, X.J.; Liu, W. An overview of mobile edge computing: Architecture, technology and direction. *KSII Trans. Internet Inf. Syst. (TIIS)* **2019**, *13*, 4849–4864.
3. Cao, K.; Liu, Y.; Meng, G.; Sun, Q. An overview on edge computing research. *IEEE Access* **2020**, *8*, 85714–85728. [CrossRef]
4. Wang, P.; Shi, H.; Yang, X.; Mi, J. Three-way k-means: Integrating k-means and three-way decision. *Int. J. Mach. Learn. Cybern.* **2019**, *10*, 2767–2777. [CrossRef]
5. Wang, P.; Yang, X. Three-way clustering method based on stability theory. *IEEE Access* **2021**, *9*, 33944–33953. [CrossRef]
6. Wang, P.; Yao, Y. CE3: A three-way clustering method based on mathematical morphology. *Knowl.-Based Syst.* **2018**, *155*, 54–65. [CrossRef]
7. Wu, T.; Fan, J.; Wang, P. An improved three-way clustering based on ensemble strategy. *Mathematics* **2022**, *10*, 1457. [CrossRef]
8. Jiang, Z.; Yang, X.; Yu, H.; Liu, D.; Wang, P.; Qian, Y. Accelerator for multi-granularity attribute reduction. *Knowl.-Based Syst.* **2019**, *177*, 145–158. [CrossRef]
9. Liu, K.; Yang, X.; Yu, H.; Fujita, H.; Chen, X.; Liu, D. Supervised information granulation strategy for attribute reduction. *Int. J. Mach. Learn. Cybern.* **2020**, *11*, 2149–2163. [CrossRef]
10. Liu, K.; Yang, X.; Fujita, H.; Liu, D.; Yang, X.; Qian, Y. An efficient selector for multi-granularity attribute reduction. *Inf. Sci.* **2019**, *505*, 457–472. [CrossRef]
11. Yang, X.; Yao, Y. Ensemble selector for attribute reduction. *Appl. Soft Comput.* **2018**, *70*, 1–11. [CrossRef]
12. Tang, B.; Luo, J.; Obaidat, M.S.; Vijayakumar, P. Container-based task scheduling in cloud-edge collaborative environment using priority-aware greedy strategy. *Clust. Comput.* **2022**, *26*, 3689–3705. [CrossRef]
13. Fan, X.; Zheng, H.; Jiang, R.; Zhang, J. Optimal design of hierarchical cloud-fog&edge computing networks with caching. *Sensors* **2020**, *20*, 1582. [PubMed]
14. Chen, X.; Zhang, J.; Lin, B.; Chen, Z.; Wolter, K.; Min, G. Energy-efficient offloading for DNN-based smart IoT systems in cloud-edge environments. *IEEE Trans. Parallel Distrib. Syst.* **2021**, *33*, 683–697. [CrossRef]
15. Dou, H.; Xu, Z.; Jiang, X.; Cui, J.; Zheng, B. Mobile edge computing based task offloading and resource allocation in smart grid. In Proceedings of the 2021 13th International Conference on Wireless Communications and Signal Processing (WCSP), Changsha, China, 20–22 October 2021; pp. 1–5.
16. Niu, X.; Shao, S.; Xin, C.; Zhou, J.; Guo, S.; Chen, X.; Qi, F. Workload allocation mechanism for minimum service delay in edge computing-based power Internet of Things. *IEEE Access* **2019**, *7*, 83771–83784. [CrossRef]
17. Sonmez, C.; Ozgovde, A.; Ersoy, C. Fuzzy workload orchestration for edge computing. *IEEE Trans. Netw. Serv. Manag.* **2019**, *16*, 769–782. [CrossRef]
18. Ali, H.S.; Rout, R.R.; Parimi, P.; Das, S.K. Real-time task scheduling in fog-cloud computing framework for iot applications: A fuzzy logic based approach. In Proceedings of the 2021 International Conference on COMmunication Systems & NETworkS (COMSNETS), Bangalore, India, 5–9 January 2021; pp. 556–564.
19. Ciric P, Z.; Stojic, D.; Sedlak, O.; Marcikic Horvat, A.; Kleut, Z. Innovation model of agricultural technologies based on intuitionistic fuzzy sets. *Sustainability* **2019**, *11*, 5457. [CrossRef]
20. Besiktepe, D.; Ozbek, M.E.; Atadero, R.A. Condition assessment framework for facility management based on fuzzy sets theory. *Buildings* **2021**, *11*, 156. [CrossRef]
21. Woźniak, M.; Szczotka, J.; Sikora, A.; Zielonka, A. Fuzzy logic type-2 intelligent moisture control system. *Expert Syst. Appl.* **2024**, *238*, 121581. [CrossRef]
22. Vlachos, I.K.; Sergiadis, G.D. Intuitionistic fuzzy information—Applications to pattern recognition. *Pattern Recognit. Lett.* **2007**, *28*, 197–206. [CrossRef]

23. Castillo, O. Optimization of type-2 and intuitionistic fuzzy systems in intelligent control. In *Uncertainty and Imprecision in Decision Making and Decision Support: New Challenges, Solutions and Perspectives, Proceedings of the BOS-2018, Warsaw, Poland, 24–26 September 2018, and Proceedings of the IWIFSGN-2018, Warsaw, Poland, 27–28 September 2018*; Springer: Cham, Switzerland, 2021; pp. 292–300.
24. Kaushal, M.; Solanki, R.; Lohani, Q.D.; Muhuri, P.K. A novel intuitionistic fuzzy set generator with application to clustering. In Proceedings of the 2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8.
25. Zadeh, L.A. Fuzzy sets. *Inf. Control* **1965**, *8*, 338–353. [CrossRef]
26. Atanassov, K.T.; Stoeva, S. Intuitionistic fuzzy sets. *Fuzzy Sets Syst.* **1986**, *20*, 87–96. [CrossRef]
27. Almutairi, J.; Aldossary, M. A novel approach for IoT tasks offloading in edge-cloud environments. *J. Cloud Comput.* **2021**, *10*, 28. [CrossRef]
28. Castillo, O.; Melin, P. A new method for fuzzy inference in intuitionistic fuzzy systems. In Proceedings of the 22nd International Conference of the North American Fuzzy Information Processing Society, NAFIPS 2003, Chicago, IL, USA, 24–26 July 2003; pp. 20–25.
29. Cherkassky, V. Fuzzy inference systems: A critical review. In *Computational Intelligence: Soft Computing and Fuzzy-Neuro Integration With Applications*; Springer: Berlin/Heidelberg, Germany, 1998; pp. 177–197.
30. Iancu, I. A Mamdani type fuzzy logic controller. *Fuzzy Log.-Control. Concepts Theor. Appl.* **2012**, *15*, 325–350.
31. Mendel, J.M. Fuzzy logic systems for engineering: A tutorial. *Proc. IEEE* **1995**, *83*, 345–377. [CrossRef]
32. Sonmez, C.; Ozgovde, A.; Ersoy, C. Edgecloudsim: An environment for performance evaluation of edge computing systems. *Trans. Emerg. Telecommun. Technol.* **2018**, *29*, e3493. [CrossRef]
33. Calheiros, R.N.; Ranjan, R.; Beloglazov, A.; De Rose, C.A.; Buyya, R. CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Softw. Pract. Exp.* **2011**, *41*, 23–50. [CrossRef]
34. Flores, H.; Srirama, S. Adaptive code offloading for mobile cloud applications: Exploiting fuzzy sets and evidence-based learning. In Proceeding of the fourth ACM Workshop on Mobile Cloud Computing and Services, Taipei, Taiwan, 25–28 June 2013; pp. 9–16.