Multiattribute Evaluation Model Based on the KSP Algorithm for Edge Computing

RUIZHONG DU\textsuperscript{1,2}, KUNQI XU\textsuperscript{1}, AND XIAOYAN LIANG\textsuperscript{1,2}

\textsuperscript{1}School of Cyber Security and Computer, Hebei University, Baoding 071002, China
\textsuperscript{2}Key Laboratory on High Trusted Information System in Hebei Province, Baoding 071002, China

Corresponding author: Kunqi Xu (jade_x86@qq.com)

This work was supported in part by the Natural Science Foundation of China under Grant 61170254, in part by the Hebei Natural Science Project under Grant F2018201153, and in part by the Hebei Natural Science Foundation Key Program Project under Grant F2019201290.

ABSTRACT To solve the problems of single evaluation attributes and highly overlapping trust paths in the current trust model, a multiattribute trust evaluation model based on the K shortest paths (KSP) algorithm is proposed. The model refines the evaluation attributes among nodes and uses the analytic hierarchy process (AHP) to allocate the weights based on users’ preferences to meet the special needs of individual users. Also, the model introduces the penalty factor algorithm idea of KSP and proposes a trust path optimization algorithm RKSP based on the \textit{A*} algorithm. It can filter highly overlapping trust paths during the formation of recommended trust paths so that the searched trust paths have certain differences. Through comparative experiments, it is proven that the model can reduce the resource overhead of edge devices, improve the accuracy of evaluation, ensure load balancing within the domain, and better align the results of the model recommendation with user needs.

INDEX TERMS Trust model, multiattribute, trust path, KSP algorithm.

I. INTRODUCTION

The rise of the Internet of Everything and 5G technology leads to new computing models such as fog computing [1], edge computing [2] and mobile edge computing [3]. Edge computing can provide near ground real-time computing and storage functions through smartphones, cameras, wearable devices, smart gateways, etc. It can reduce the pressure of the cloud computing center and network bandwidth, with the advantages of low latency and high energy efficiency [4], [5]. Moreover, it has a good application effect in intelligent car networking, virtual reality, healthcare, smart homes, smart cities, industrial Internet of Things and other scenarios [6]–[8]. However, the characteristics of edge devices, such as openness, dynamics, autonomy, and the lack of stable basic protection measures, result in the lack of necessary trust between devices. As a result, it is difficult for devices to resist unreliable fraud and services, the collaboration between devices is prone to problems, and considerable private data can become exposed [9], [10].

Establishing trust relationships between devices to detect untrusted entities can effectively eliminate malicious and selfish nodes, reduce the risk of independent decision-making, and improve system reliability [11]. Compared with the cloud computing environment, the traditional centralized security mechanism is no longer suitable for the fully distributed edge computing architecture. The edge layer contains a large number of high-frequency interactive devices, and the number of nodes and trust information is growing exponentially. While edge layer devices are mostly resource-constrained devices, massive trust information easily leads to information overload, causing node overload and early decline [12], [13]. A trusted and lightweight distributed trust evaluation mechanism is urgently needed. Therefore, the trust model in the edge computing environment has gradually become a research hotspot.

In the face of limited resources and open edge computing, many experts and scholars have adopted different methods for building trust evaluation models for edge computing environments [14]–[17]. In addition to the evaluation model based on subjective logic, Dempster-Shafer evidence theory, Bayesian networks and other trust evaluation models, there are also models based on recommendation node similarity, scoring deviation mixed methods, multiattribute evaluation, and so on. Next, we will discuss the following three aspects:

In terms of improving effectiveness, Deng \textit{et al} [18] proposed a multiobjective optimization and collaboration scheme based on comprehensive trust, which optimized the
edge computing resource management and collaboration system by using the trust evaluation system, and then improved the accuracy of the model. Huang et al [19] weighted different trust dimensions according to familiarity, similarity and timeliness and then maintained and updated the trust information of local vehicles by using a vector machine and multiweight subjective logic. He et al [20] combined the Bayesian reasoning method and D-S evidence theory and optimized the uncertainty in the trust evaluation of mobile social networks by using a deep learning algorithm. It could significantly reduce the adverse impact of biased opinions. However, the above model did not consider the unreliability of the trust model and ignored the possibility of malicious recommendation. To this end, Ruan et al [21] proposed a trust management framework based on measurement theory, which regards the measurement error of trust evaluation as confidence, measures the reliability of equipment trust evaluation, and improves the accuracy of the trust value. Ren et al [22] introduced the blockchain consensus mechanism to prevent trust data from being forged and tampered with. This model provided differentiated trust management options for devices with different computing and storage capabilities, which did not rely on trusted third-parties and interdomain trust assumptions.

In terms of reducing resource overhead, Jie et al [23] proposed a trust evaluation model based on multisource feedback, which added feedback trust from the base station to the traditional trust relationship. Additionally, the entropy weight method was used to aggregate the multisource feedback trust to enhance the adaptability of the trust model, but malicious feedback information was not filtered. Gao et al [24] proposed a dual filtering K-means clustering algorithm, which effectively filtered the feedback of low similarity and malicious devices in the current task context and improved the computing efficiency while resisting malicious attacks. Kammoun et al [25] regarded the base station at the edge of the network as a trusted third-party and proposed a single-hop clustering mechanism based on node density, trust and node energy level. In this scheme, the energy consumption of nodes is fully considered, and the resource consumption of trust computing is reduced.

In the multiattribute evaluation model, Ma et al [26] proposed a trust evaluation model based on multi service attributes for the cloud service environment. The service requester determines the transaction object by integrating multiple service attributes. Then, according to the actual service quality, the corresponding rewards or punishments value will be given. However, the model is only a simple framework that does not have specific attributes and attribute weights. For the threats in mobile ad hoc network (MANETs) Khan et al [27] set the generation and packet loss rate of control packets as the attributes of nodes, and compared the trust value with the threshold value to determine the credibility of nodes. However, the attribute weight is directly given by the model based on the importance of the current time attribute, which is not sufficiently rigorous. Ma et al [28] proposed a model based on integrated trust. The attributes are divided into public attributes and trust attributes. This model pays more attention to the identity trust of nodes, and does not emphasize the behavior attribute of nodes very well.

These researches work effectively promoting the development of trust evaluation models in edge computing and enhanced the reliability of the system. However, most of the current trust models still have the following shortcomings:

1. The existing multiattribute trust evaluation model does not consider the multiattribute problem of behavior trust and cannot reflect the subjectivity and complexity of the trust relationship between devices.

2. The existing trust evaluation model based on graph theory ignores the deviation of evaluation results, which is caused by the high overlap of trust paths. This model has difficulty in resisting collusion attacks between devices and also increases the calculation costs.

To solve the above problems, this paper introduces the K shortest paths (KSP) algorithm and proposes a multiattribute evaluation model based on the KSP optimization algorithm. First, the multiattribute trust evaluation model is constructed to refine the evaluation attributes, and the attribute weight is determined by using the analytic hierarchy process (AHP). Second, to reduce the recommendation of the same node during the formation of the recommended trust path, the trust path optimization algorithm RKSP is proposed to solve the trust dependence problem, and it filters highly overlapping trust paths based on the A* algorithm and penalty factor idea. Finally, experiments verify that the model can overcome the load balance problem of the current trust model and improve the efficiency of trust calculations.

The remainder of the paper is organized as follows. In Section 2 we introduce the architecture and working principle of the trust evaluation model for edge computing. We detail the multi-attribute trust evaluation model and define the trust relationship between devices in Section 3. In Section 4, we present our RKSP algorithm based on the penalty factor. We describe experimental settings and analyze our experimental results in Section 5. The conclusions are in Section 6.

II. SYSTEM ARCHITECTURE OF THE TRUST EVALUATION MODEL FOR EDGE COMPUTING

Edge computing is a distributed structure, which is a decentralized computing architecture. It moves applications, user data and various services from the original network center node to the edge node in the network logic for processing and provides near-earth storage, computing and other functions [9].

The trust model architecture based on edge computing consists of the edge layer into smaller and more manageable areas for management, which is called the management domain (MD). It transforms the original cloud computing center processing into edge server (ES) processing. Each MD consists of an ES and its subordinate edge devices (ED). The ES is responsible for creating, evaluating and updating the trust relationship...
in the region, ensuring the operation of the trust model and
the accuracy of the trust evaluation results. The EDs are
divided into different ES according to their location and
characteristics and are managed by the servers. To realize
the sharing of trust information, the devices can dynamically
adjust and update the trust relationship with other devices.

![System architecture of the trust evaluation model based on
dge computing.](image)

III. MULTIATTRIBUTE TRUST EVALUATION MODEL

In practical applications, the expected service attributes
of different ED are different; these different needs and
preferences ultimately lead to different evaluation results
in the services provided by the same object device across
different main devices. For example, in the task of video
caching, devices focus more on the speed of video caching
than download quality, download cost and other factors.
Therefore, interactive devices with fast response times obtain
higher trust values. However, this does not mean that the
device will perform well in completing the service tasks
required by other devices. Therefore, in the process of
evaluation, the equipment needs to consider the completion
of the task from many aspects, such as response time,
execution cost, reliability, availability, etc., to provide a more
appropriate evaluation.

Accordingly, this section constructs a trust evaluation
model based on multiple attributes, and gives the description
and calculation equation of the trust relationship between
devices.

Assume that the all-devices set in the domain is $ED = \{ed_1, ed_2, ed_3, \ldots, ed_n\}$, and the multiattribute evaluation set
is $ATTR = \{attr_1, attr_2, attr_3, \ldots, attr_m\}$. Equipment $ed_i$ is
measured according to attribute $attr_j$, and attribute value $o_{ij}$
of $ed_i$ about $attr_j$ is obtained. Then, the following decision
matrix is formed:

$$O = \begin{pmatrix}
o_{11} & o_{12} & \cdots & o_{1m} \\
o_{21} & o_{22} & \cdots & o_{2m} \\
\vdots & \vdots & \ddots & \vdots \\
o_{i1} & o_{i2} & \cdots & o_{im}
\end{pmatrix} \tag{1}
$$

Because different attributes have different physical mean-
ings and value ranges, it is difficult to directly carry out a
comparative analysis; therefore, it is necessary to perform
standardization. In this paper, the range transformation
method is used to deal with the cost type attribute and the
benefit type attribute by equation (2), and the final decision
matrix $R$ is obtained.

The equation of matrix element $r_{ij}$ is:

$$r_{ij} = \begin{cases}
o_{ij}^\text{max} - o_{ij} & \text{if } attr_j \text{ is benefit type attribute} \\
\frac{o_{ij}^\text{max} - o_{ij}^\text{min}}{o_{ij}^\text{max} - o_{ij}^\text{min}} & \text{if } attr_j \text{ is cost type attribute} \\
\end{cases} \tag{2}
$$

where $o_{ij}^\text{max}$ represents the maximum value of matrix $O$,
$o_{ij}^\text{min}$ represents the minimum one, $i$ represents the equipment
number, and $j$ represents the attribute number. In addition,
$attr_j$ represents the current attribute.

The final standardized multiattribute matrix is:

$$R = \begin{pmatrix}
r_{11} & r_{12} & \cdots & r_{1m} \\
r_{21} & r_{22} & \cdots & r_{2m} \\
\vdots & \vdots & \ddots & \vdots \\
r_{n1} & r_{n2} & \cdots & r_{nm}
\end{pmatrix} \tag{3}
$$

A. DIRECT TRUST

Direct trust is the trust evaluation obtained by the device
based on its own historical interaction information and the
result of aggregating multiple attributes.

To weigh the influence of different attributes on the
device trust value, it is necessary to determine the weight of
each attribute, and different devices have different
weight vectors $W_i = [w_1, w_2, w_3, \ldots, w_m]$. In multi-
attribute decision-making problems, weight determination
methods include the entropy method [29], fuzzy clustering
method [30], principal component analysis method [31], and
analytic hierarchy process (AHP) [32]. In this paper, AHP is
used to confirm the weight, and the specific operation process
is as follows:

1. According to the nine-scale method (Table 1), the two
attributes are compared to obtain the weight judgment matrix
$A = (a_{ij})_{m \times m}$, and the element $a_{ij}$ indicates the importance
of attribute $attr_j$ over attribute $attr_i$ under the subjective
judgment of the equipment, where $a_{ii} = 1$, $a_{ij} = \frac{1}{a_{ji}}$. 



146934

VOLUME 8, 2020
(2) According to the weight judgment matrix, the weight equation of attribute $attr_i$ is:

$$w_i = \frac{1}{m} \left[ \prod_{j=1}^{m} a_{ij} \right]$$

where $w_i$ is the weight of attribute $attr_i$, $a_{ij}$ and $a_{ij}$ are the elements of weight judgment matrix $A$, and $m$ is the number of attributes.

(3) Because the importance of attributes is based on the subjective judgment of users, it may not meet the consistency principle required by AHP, so the consistency ratio of the matrix needs to be tested.

The equation of consistency ratio $CR$ is as follows:

$$CR = \frac{CI}{RI}$$

where $CR$ is the consistency ratio, $CI$ is the consistency indicators, and $RI$ is the average random consistency index, and the standard value of $RI$ is given in Table 2. $\lambda$ is the maximum eigenvalue of matrix $A$, $m$ is the number of attributes, and $w_i$ is the weight of attribute $attr_i$.

### Table 1. Nine-level scale method [32].

| Meaning                  | Scale |
|--------------------------|-------|
| Equal importance         | 1     |
| Moderate importance      | 3     |
| Strong importance        | 5     |
| Demonstrated importance  | 7     |
| Extreme importance       | 9     |
| Intermediate values      | 2, 4, 6, 8 |

![Figure 2. Recommended trust.](image)

**B. RECOMMENDED TRUST**

Recommendation trust is a trust evaluation based on the aggregation of trust opinions of other trusted neighbors. As shown in Figure 2, the recommended trust is passed through the nodes that interact with $V_i$ or $V_j$, that is, the path from $V_i$ to $V_j$. Each trust path corresponds to a recommended trust value, reflecting a recommended trust relationship between nodes. It takes into account the feedback of multiple recommended trust paths and can describe the objective credibility.

According to the trust transfer and aggregation rules, the recommended trust calculation equation is as follows:

$$F_{ij} = \frac{1}{k} \sum_{x=1}^{k} P_{i \rightarrow j}^{x}$$

where $F_{ij}$ refers to the aggregate value of recommended trust of all trust paths between two nodes; that is, the global recommended trust of node $V_i$ to node $V_j$, and $k$ represents the total number of trust paths between nodes. $P_{i \rightarrow j}^{x}$ refers to the recommended trust value calculated from the x-th trust path.

### Table 2. RI value [32].

| $m$ | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| $RI$| 0.00| 0.58| 0.90| 1.12| 1.24| 1.32| 1.41| 1.45|

According to the regulations, when the consistency ratio $CR$ is less than 0.1, it is considered that the matrix has passed the consistency test and the attribute weight can be calculated accordingly. If not, the weight judgment matrix should be modified and then tested.

In summary, let $W_i = [w_1, w_2, w_3, \cdots w_j]^T$ be the weight vector of the current node, where $w_j$ is the weight of attribute $attr_j$ and satisfies $\sum w_j = 1 (0 \leq w_j \leq 1)$. The comprehensive evaluation of the equipment is as follows:

$$I_i = \sum r_{ij}w_j$$

**C. COMPREHENSIVE TRUST**

Comprehensive trust is the global trust of a device to another device, which is the final trust value obtained by aggregating the direct trust and recommended trust in some way. When there is no direct interaction record between nodes, the recommended trust is regarded as the comprehensive trust to build the trust between unfamiliar nodes. To improve the reliability of trust, and overcome the limitation of subjective assignment, an adaptive aggregation method based on information entropy theory is used.

Entropy weight method uses the difference between information to weight attributes, and effectively corrects the degree of difference between trust values, which is more objective. However, while using this method, we need to have a certain sample size to determine the weight through the sample.
Based on the theory of information entropy, the comprehensive trust $T_{ij}$ is calculated as follows:

$$T_{ij} = \omega_1 \times I_{ij} + \omega_2 \times F_{ij}$$  \hspace{1cm} (11)

where $\omega_1$ and $\omega_2$ are the adaptive weights of direct trust and feedback trust, respectively, and their calculation equation is as follows:

$$\omega_1 = \frac{1 - H(I_{ij})}{\log_2 I_{ij}}$$  \hspace{1cm} (12)

$$\omega_2 = \frac{1 - H(F_{ij})}{\log_2 F_{ij}}$$  \hspace{1cm} (13)

where $H(D_{ij})$ and $H(F_{ij})$ are the information entropy of direct trust and feedback trust, respectively, and their calculation equation is as follows:

$$H(I_{ij}) = -I_{ij} \log_2 I_{ij} - (1 - I_{ij}) \log_2 (1 - I_{ij})$$  \hspace{1cm} (14)

$$H(F_{ij}) = -F_{ij} \log_2 F_{ij} - (1 - F_{ij}) \log_2 (1 - F_{ij})$$  \hspace{1cm} (15)

IV. RKSP OPTIMIZATION ALGORITHM BASED ON A PENALTY FACTOR

In a large-scale trust graph, especially in the case of a large number of nodes and frequent interaction in the edge computing environment, the number of trust paths between two nodes is usually very large. The search and aggregation of trust paths require considerable time and space resources. If all trust paths between two nodes are exhausted, the calculation cost will increase. In addition, there may be a large number of shared nodes or shared edges in multiple trust paths between two nodes that will lead to an overreliance of the obtained trust information on the recommendation value of some nodes. Furthermore, collusion attacks are easily caused between malicious nodes and make the nodes misjudge recommendations. Therefore, this paper optimizes the trust path by limiting the number of trust paths and the repetition between paths.

The trust path from source node S to destination node E is shown in Table 3.

TABLE 3. Trust path from S to E.

| Path |     |
|------|-----|
| $P_1$ | S→1→4→⋯→E |
| $P_2$ | S→1→5→⋯→E |
| $P_3$ | S→1→6→⋯→E |
| $P_4$ | S→2→6→⋯→E |
| $P_5$ | S→3→7→⋯→E |

From Table 3, it can be seen that the first three of the five trust paths from node S to node E pass through the trust edge $S\rightarrow 1$, indicating that these paths all depend on the recommendation information of node 1. If node 1 is a malicious node participating in collusive attack [33], then it can cheat the source node S by providing recommendation information that denigrates honest nodes and exaggerates similar nodes. Therefore, the recommendation information obtained by node s is no longer reliable.

When the first k paths selected by the model are derived from several nodes, the final trust value will depend on the trust recommendation of these nodes. Therefore, the whole trust network is controlled by these nodes. Therefore, the highly overlapping trust paths of nodes or trust edges will greatly affect the multisource trust recommendation, not only rendering it difficult to resist collusion attacks between nodes but also leading to an unreliable trust evaluation model.

Although the KSP algorithm can reduce the search time of the trust path, it will also inadvertently increase the risk of the trust model if it does not restrict the path search method. To overcome the limitations of the traditional KSP algorithm, the paper adds constraints in the search process of the trust path and finally obtains multiple trust paths with differences.

A. KSP ALGORITHM PROBLEM DESCRIPTION

Given a trust graph G, there are multiple different trust paths from one node $v_i$ to another node $v_j$, and the resulting set $\text{paths}(G, v_i, v_j) = \{\text{path}_1, \text{path}_2, \text{path}_3, \ldots, \text{path}_k\}$ ($k \in N^*$) is called the path set between $v_i$ and $v_j$ on G. The top k paths between $v_i$ and $v_j$ on the trust graph G are found, and the repeatability of any two paths that meets the requirements is determined, which is called the problem of the top k shortest paths that meet a certain degree of repetition.

B. RKSP ALGORITHM

Because the lossy algorithm in the KSP algorithm has the advantage of computing efficiency, it can meet the needs of a large-scale trust graph in the edge environment. Therefore, based on the idea of the penalty factor in the lossy algorithm, by introducing the repeatability limit factor and A* algorithm, this paper proposes a trust subgraph optimization algorithm (RKSP) that satisfies the repeatability constraint.

The algorithm idea of the penalty factor is as follows: First, the Dijkstra algorithm is used to obtain the shortest path.
between two nodes, and the edge of the path is penalized, i.e., the weight of the edge is multiplied by the penalty factor. Then, the algorithm cycle Dijkstra algorithm is used to find the remaining path. After the penalty, the weight of the edge in the graph changes, which can prevent the edge from being traversed in the next search to reduce the repetition of the path.

According to the characteristics of the ED and trust recommendations, this paper improves the KSP algorithm as follows:

1) The scope of the penalty is extended to all derived edges of nodes contained in the current path;
2) The Dijkstra algorithm is used only when searching the first path, and the remaining paths are searched by the A* algorithm.

The above improvements can effectively reduce the proportion of the recommended value of the node in all the recommended trust paths and improve the reliability of the evaluation model. Moreover, the A* algorithm can locate the next-hop node quickly, reduce the number of access nodes, and reduce the search cost of the algorithm.

1) A* ALGORITHM
The A* algorithm is a direct search algorithm in the shortest path algorithm that is suitable for complex large-scale graphs. By setting a heuristic function, the path searching process can be more directional and faster. The A* algorithm uses equation (17) to search the path:

\[
f(v) = g(v) + h(v)
\]

where \( f(v) \) represents the total cost when the path passes through node \( v \), \( g(v) \) represents the actual cost from the source node to the current node \( v \), and \( h(v) \) represents the estimated cost from the current node \( v \) to the destination node. As the heuristic function of the algorithm, \( h(v) \) is more accurate, and its efficiency is better.

2) HEURISTIC FUNCTION
The heuristic function \( h(v) \) is the distance set \( D \) from each node to the source node. Set \( D \) can be obtained by the Dijkstra algorithm and changes with the penalty for the trust path. The penalty equation is as follows:

\[
D_v = D_v \times \alpha
\]

where \( \alpha \) is the penalty factor, and the specific value depends on the node tolerance.

3) REPEATABILITY FUNCTION
The repeatability function is used to calculate the repeatability between paths and add the trust paths that meet the repeatability requirements to the path set. Assume there are two trust paths \( path_i \) and \( path_j \), where the repetition of \( path_i \) with respect to \( path_j \) is equal to the ratio of the number of the same trust edges in the two paths to the number of all trust edges in the trust path, as shown in equation (18).

Set the threshold value of repeatability to \( \theta \), and the value of \( \theta \) can be adjusted according to the user’s tolerance.

\[
repetition(path_i, path_j) = \frac{|SameEdge(path_i, path_j)|}{|path_i|}
\]

The pseudocode of the RKSP algorithm is shown in Algorithm 1.

**Algorithm 1**

**Input:** Trust graph \( G(V, E) \), source node \( s \), destination node \( e \)

**Output:** Path set \( Pset \)

1. Dijkstra\((G, s, e)\)
2. get a path \( p \) and a set \( D \)
3. \( Pset \leftarrow Pset \cup p \)
4. for \( v \in p \) do
   5. \( D_v = D_v \times \alpha \)
6. while \( k < K_{th} \) do
   7. if \( c > \text{count} \) then
      8. \( \theta = \theta + \Delta \theta, \ c = 0 \)
   9. for \( v \in \text{openlist} \) do
      10. if \( v == e \) then
          11. go to line 19
      12. if \( v \in \text{closed} \) then
          13. \( v = v \rightarrow \text{next} \)
      14. \( u = \text{min}(f) \)
      15. \( v = \text{father}(u) \)
      16. add Dst\((u)\) in openlist
      17. add \( u \) in closed
      18. remove \( u \) from openlist
      19. while \( v == s \) do
          20. \( v = \text{father}(v) \)
          21. \( p = p \cup \text{father}(v) \)
      22. if \( \text{Repetition}(p, pset) < \theta \) then
          23. \( Pset \leftarrow Pset \cup p \)
          24. \( k = k + 1 \)
      25. for \( v \in p \) do
          26. \( D_v = D_v \times \alpha \)
27. return \( Pset \)

where \( \text{openlist} \) represents the set of waiting access nodes, \( \text{closed} \) represents the set of visited nodes, \( p \) represents the current trust path, \( Pset \) represents the set of trust paths, \( K_{th} \) represents the number of paths to be found, \( \text{father}(u) \) represents the parent of node \( u \), and \( \text{Dst}(u) \) represents the adjacent nodes of node \( u \). First, the Dijkstra algorithm is used to preprocess the trust graph, the shortest path \( p \) from the source node to the destination node and the shortest path distance set \( D \) from each node to the destination node are obtained. Then, distance set \( D \) is used as the heuristic information of the A* algorithm, and the A* algorithm is called to search the remaining paths. Additionally, the repeatability function is used to determine whether the path meets the repeatability requirements until \( K_{th} \) trust paths are finally obtained. When the number of paths is less than \( K_{th} \)
at $\theta$, the limit of repeatability can be relaxed by setting the upper limit $count$ of cycle times and the increasing factor $\Delta \theta$ of repeatability. It can avoid the algorithm falling into a dead cycle until the path reaches $Kth$ or the upper limit of repeatability.

The penalty factor can be used to make the trust path after each cycle different from the last traversal result. In addition, the search of the A* algorithm is directional, which can greatly reduce the number of access nodes and improve the efficiency of the trust path search. Therefore, it can make recommendation information more multisource and, closer to the actual situation.

V. SIMULATION EXPERIMENT

To verify the reliability and resource cost of the model, MATLAB is used in this paper to carry out simulation experiments that are compared with the RLTS (reliable and lightweight trust scheme) model [23] and the RFSN model [34]. The simulation detection area was $200 \times 200$ square, and 200 nodes were randomly placed to simulate the ED with limited resources.

To make the experiment closer to the real edge computing environment, the nodes were divided into the following categories:

1) Honest nodes were 90% likely to provide high-quality services and simulate node anomalies caused by nonintrusive factors;
2) Honest nodes had three states: idle, normal, and busy. In different states, honest nodes had different probabilities of refusing service requests, which were 10%, 20%, and 40%. Honest nodes simulated the ED with different degrees of being busy. For a node, the probabilities of the above states were 50%, 30%, and 20%.
3) Malicious nodes were divided into two categories: one provided malicious services and provided dishonest recommendation information to other nodes, and the other provided honest services and provided recommendation information that denigrated honest nodes and exaggerated similar nodes. The above nodes accounted for half of each;
4) All nodes were divided into three categories: fast, general, and delay. Different types of nodes had different response times for the same task. The nodes simulated the ED with different collaboration speeds. The proportions of the above nodes were 50%, 30%, and 20%.

A. PROPERTY SETTINGS

According to the characteristics of the ED, the following attributes were selected to evaluate the device in the experiment: reliability ($R_1$), availability ($R_2$), response time ($R_3$), and node residual energy ($R_4$).

1) Reliability ($R_1$): the ability of the device to complete interactive tasks that is described by the success rate of the service. The node reliability equation is as follows:

$$R_1 = \frac{c^s + 1}{c^{accept} + 2}$$  \hspace{1cm} (19)

where $c^s$ is the number of successful interactions, and $c^{accept}$ is the number of times to accept interactive tasks.

2) Availability ($R_2$): the ability of a device to respond to task requests. When the device is suspended due to failure, being busy and other reasons, it cannot respond to the service request of other devices. The node availability equation is as follows:

$$R_2 = \frac{c^{accept} + 1}{c^{apply} + 2}$$  \hspace{1cm} (20)

where $c^{accept}$ is the number of times to accept interactive tasks, and $c^{apply}$ is the total number of times to apply for tasks.

3) Response time ($R_3$): the time from the service request to task completion that is provided by the device to the edge server.

4) Node residual energy ($R_4$): the current energy value of the node that is provided to the ES by the node to be evaluated. It reflects certain subjectivity.

Since the attributes $R_1$ and $R_2$ are the values of $[0, 1]$, only attributes $R_3$ and $R_4$ needed to be normalized by equations (2) and (3). In addition, attributes $R_2$, $R_3$, and $R_4$ have no delivery function and cannot be delivered through the trust path. Therefore, only attribute $R_1$ (node reliability) was delivered for trust, and other attributes were recommended for calculation at the end of the recommendation.

B. PARAMETER SETTING

1) K-VALUE

The source node and the destination node were randomly selected from 200 nodes to obtain the trust subgraph between the two nodes. The change degree of the recommended trust value between nodes was observed when the $k$-value gradually increased, as shown in Figure 4.

![FIGURE 4. Influence of the k-value on the trust value.](image-url)
10%, the trust value tended to be stable when the number of paths reached 20. Moreover, with the increase in the number of malicious nodes, the k-value increased when trust value reached a stable state.

Considering the different malicious ratios, we set the k-value to 40 for the subsequent experiments. In fact, different trust subgraphs may lead to a change in the k-value, but the fluctuation is not too large. This experiment was performed solely to establish a reference.

2) PENALTY FACTOR $\alpha$

The setting of the penalty factor should not only consider the difference degree of the trust path obtained in the next cycle but also consider that the trust value cannot be lower than the untrusted node after the penalty. The initial trust value of the trust side was set to 1 as an experimental example. The results are shown in Figure 5.

![FIGURE 5. Influence of the penalty factor on the value of the trust edge.](image)

As seen from Figure 5, the larger the value of $\alpha$ was, the stricter the limit on the number of repetitions of the trust edge. When the value of $\alpha$ was 1.1, the influence on the trust value was too small to constrain the repetition. When the value of $\alpha$ was greater than 1.4, the trust value was lower than the middle value of 0.5 after two penalties. For the same $\alpha$ value, the lower the initial trust value was, the faster the trust value decreased, and the number of repetitions that could be tolerated also decreased. Therefore, the value with a higher tolerance was selected in this paper and the value of $\alpha$ was set to 1.2. In fact, the value of the penalty factor can be adjusted according to the tolerance of nodes.

3) REPETITION LIMIT FACTOR AND REPETITION INCREASING FACTOR

For the repeatability limit factor $\theta$, the smaller the initial value was, the more stringent the repeatability limit of the trust edge. However, this approach also easily caused the number of paths to be less than $k$, which made the limiting factor increase continuously and increased the complexity of the algorithm. Similarly, if the repetition increasing factor $\Delta \theta$ was too small, the tolerance could not be increased. If it was too large, the tolerance could increase too much. The relationship between repeatability and path length is shown in Table 4.

| Path length | Repeat 1 edge | Repeat 2 edge | Repeat 3 edge |
|-------------|---------------|---------------|---------------|
| 2           | 0.5           | 1             | -             |
| 3           | 0.333         | 0.667         | 1             |
| 4           | 0.25          | 0.5           | 0.75          |
| 5           | 0.2           | 0.4           | 0.6           |
| 6           | 0.167         | 0.333         | 0.5           |
| 7           | 0.143         | 0.286         | 0.429         |
| 8           | 0.125         | 0.25          | 0.375         |
| 9           | 0.111         | 0.222         | 0.333         |
| 10          | 0.1           | 0.2           | 0.3           |

In practice, because the average path length of the sparse matrix is different from that of the dense matrix, it is impossible to measure the path repeatability with a fixed value. Starting from a stricter repeatability and then using the increasing function to gradually relax the repeatability can be considered. In this paper, the limiting factor $\theta$ and repetition increasing factor $\Delta \theta$ of repeatability were set to 0.2.

C. RELIABILITY EVALUATION

1) DIFFERENTIATION OF PERSONALITY PREFERENCE

The weight was calculated by AHP. It was assumed that the preference degree of device A for each attribute was reliability > availability > response time > node residual energy.

According to the nine-scale method, the following matrix can be obtained:

$$A = \begin{pmatrix}
1 & 3 & 3 & 5 \\
1/3 & 1 & 2 & 3 \\
1/3 & 1/2 & 1 & 2 \\
1/5 & 1/3 & 1/2 & 1
\end{pmatrix}$$

Through calculation, the consistency ratio $CR = 0.0219 < 0.1$ was obtained, which met the consistency requirements. The weight vector $W = \begin{pmatrix}0.5215,0.2395,0.153,0.086\end{pmatrix}^\top$ was obtained.

This experiment mainly verified whether the trust model can better reflect the influence of subjective preference on the trust value after introducing a multiattribute evaluation mechanism. This model was compared with the trust model that relies only on reliability evaluation and the trust model that used the average weighting method, and its performance in distinguishing personality preferences was compared. The experimental results are shown in Figure 6.

It can be seen from Figure 6 that due to its coarse granularity, the evaluation model of a single attribute no longer had a large change after the trust value tended to be...
stable. After the introduction of the multiattribute evaluation model, due to the influence of each attribute, the trust value had a significant change with the increase in interaction times. The two weighted methods reflected the influence of different weight vectors on the trust value of nodes. The degree of fluctuation of the trust value obtained by AHP was more obvious, which showed that the method can better distinguish the subjective bias of the node without affecting the overall trend of the trust.

2) ANALYSIS OF THE INTERACTION SUCCESS RATE

The successful interaction rate refers to the ratio of the number of successful interactions between devices to the total number of interactions, which is used to measure whether the trust model can effectively resist the fraud of malicious nodes.

To verify the reliability of the scheme, 100 interaction cycles of simulation experiments were carried out to investigate the change in the interaction success rate of the trust model with the increase in malicious node proportion. Each node initiated a service request in each cycle, and the node could be both a service requester and a service provider. The change in the interaction success rate with the increase in malicious nodes is shown in Figure 7.

With the increase in the proportion of malicious nodes, the interaction success rate of each trust model showed a downward trend to varying degrees. However, the interaction success rate of the model in this paper had little difference, and the curve was relatively flat. This is because, in the process of obtaining recommended trust, the RKSP algorithm filters highly overlapping trust paths by penalizing the generation value of repeated nodes, which makes the trust path more different and can effectively filter the recommendation value of malicious nodes to better reflect the behavior of nodes.

3) ACCURACY EVALUATION OF TRUST MODEL

Due to the influence of many uncertain factors, measurement error will inevitably exist in the process of trust evaluation. Model accuracy refers to the degree of agreement between the measured value and the real value, which is usually measured by error. The smaller the error, the higher the accuracy.

To measure the accuracy of trust model, this paper uses the mean absolute deviation (MAD) to calculate the error. The calculation equation is as follows:

$$\text{MAD} = \frac{\sum_{j=1}^{N_1} \sum_{i=1}^{N_2} |T_{ji} - A_i|}{N_1 N_2 NP}$$

where $T_{ji}$ is measured trust value of node $j$ to node $i$, $A_i$ is actual trust value of node $i$, and $NP$ is the total number of entity pairs with trust relationships. $N_1$, $N_2$ represent the service requesters set and service providers set.

In the simulation environment, 30% malicious nodes are randomly selected to compare the accuracy with other model. The experimental results are shown in Figure 8.

As can be seen from Figure 8, the MAD values of all models were relatively higher at the beginning, and decreased...
with the increase of cycles, which finally reached a stable state. The main reason was that the behavior of collusive malicious nodes was relatively hidden, and it was difficult for honest nodes to identify such malicious nodes in a short time. However, the MAD value of the model in this paper could quickly reach the stable state, and it was always the smallest. This is because the RKSP algorithm could filter repeated trust paths quickly. Also, the punishment of malicious nodes and the aggregation method based on information entropy could make the trust evaluation more accurate.

D. TIME COST
The time cost of the trust model mainly comes from the calculation of the comprehensive trust value, so the total time of the comprehensive trust aggregation is used to evaluate the calculation efficiency of the whole model in this paper. In the simulation environment, 30% of malicious nodes were randomly selected, and each group of experiments was conducted 5 times to obtain the average value. Figure 9 shows the time overhead change from 100 nodes to 1,000 nodes.

![Comparison of time overhead changes.](image1)

In Figure 9, when the number of nodes was small, the time cost between models was very close, but with the increase in network scale, the difference between models began to increase. In this paper, the growth rate of the time cost was relatively slow, and it was gradually lower in the proposed model than in the other two models. The reason for this was because, in the process of obtaining recommended trust, the RKSP algorithm used the KSP algorithm to reduce the number of search paths. Then, the highly overlapping trust paths were eliminated through the limit of repeatability, and the number of paths participating in the calculation was reduced. In addition, the A* algorithm could improve the search efficiency and greatly reduce the workload of trust computing.

E. ENERGY COST
Considering the energy cost, the model is verified from the energy change in the network in the domain and the node survival. Assume that the initial energy of each node is 0.5 J, and the energy consumption of each transmission and reception of data packets is 50 nJ/bit, where J represents Joule and the unit of data packets is a bit. Any data in the range [3,000, 4,000] was selected, and the size of three model packets was the same in the same interaction cycle. There were 1,000 simulation cycles in the experiment. The change in network energy in the region is shown in Figure 10, and the number of network nodes surviving is shown in Figure 11.

![Comparison of transmission energy consumption.](image2)

![Number of nodes alive.](image3)

With the increase in the number of interactions between nodes, the network energy of each model showed a downward trend. The energy consumption curve of the proposed model was relatively flat. The overall network energy of this model was also higher than that of the other two models. Because the recommended trust was stored only in the ES, it completed the trust recommendation process without the participation of the resource-constrained node. The node needed only to send the trust information and query request to the server. Therefore, the model could effectively reduce the energy consumption in the process of trust value transmission and improve the network life cycle.

As shown in Figure 11, with the increase in interaction times, each model gradually exhibited node decay to different
degrees. The RLTS model and RFSN model exhibited node death after 500 rounds and all nodes died after 2,000 and 2,500 rounds, respectively. In the proposed model, node death began after 1,500 rounds. In this paper, the energy of nodes was added to the multiattribute set. In the case of similar values of other attributes, the nodes with high energy were preferred for interaction, which could avoid the premature failure of nodes with high trust values. This shows that the multiattribute model can guarantee the load balance in the domain and prolong the lifetime of the whole network to a certain extent.

VI. CONCLUSION
With the rise of edge computing, the relationship between edge devices was becoming more complex, and the quantity of data in the network was increasing daily. It was difficult for edge devices to undertake complex storage and trust aggregation tasks. However, the quantity of data was not equal to the amount of information, which would result in deviations of the trust value and increase the computing load of the node. Therefore, based on the traditional trust evaluation model, a multiattribute evaluation model based on the KSP optimization algorithm was proposed. First, based on the multiattribute of service, the users’ preference was fully considered. Second, the KSP algorithm and A* algorithm were introduced into the field of trust evaluation, which not only ensured the efficiency of calculation but also improved the reliability of the model. The experimental results showed that this method could improve the success rate of interaction, effectively inhibit the fraud of malicious entities, and ensure the honesty of the edge device interaction.

In future research, we will consider the evaluation method for an unknown weight and evaluate more attributes to enhance the objectivity of trust model.

REFERENCES

[1] L. M. Vaquero and L. Rodero-Merino, “Finding your way in the fog: Towards a comprehensive definition of fog computing,” ACM SIGCOMM Comput. Commun. Rev., vol. 44, no. 5, pp. 27–32, Oct. 2014.

[2] M. Satyanarayanan, “The emergence of edge computing,” Computer, vol. 50, no. 1, pp. 30–39, Jan. 2017.

[3] K. Zhang, Y. Mao, S. Leng, Q. Zhao, L. Li, X. Peng, L. Pan, S. Maharjan, and Y. Zhang, “Energy-efficient offloading for mobile edge computing in 5G heterogeneous networks,” IEEE Access, vol. 4, pp. 5896–5907, 2016.

[4] W. Shi and S. Dusdjar, “The promise of edge computing,” Computer, vol. 49, no. 5, pp. 78–81, May 2016.

[5] W. Z. Khan, E. Ahmed, S. Hakak, I. Yaqoob, and A. Ahmed, “Edge computing: A survey,” Future Gener. Comput. Syst., vol. 97, pp. 219–235, Aug. 2019.

[6] P. Pace, G. Aloisio, R. Gravina, G. Caliciuri, G. Fortino, and A. Liotta, “An edge-based architecture to support efficient applications for healthcare industry 4.0,” IEEE Trans. Ind. Informat., vol. 15, no. 1, pp. 481–489, Jan. 2019.

[7] J. M. Batalla and F. Gonciarz, “Deployment of smart home management system at the edge: Mechanisms and protocols,” Neural Comput. Appl., vol. 31, no. 5, pp. 1301–1315, May 2018.

[8] X. Li, Z. Zhou, J. Guo, S. Wang, and J. Zhang, “Aggregated multi-attribute query processing in edge computing for industrial IoT applications,” Comput. Netw., vol. 151, pp. 114–123, Mar. 2019.

[9] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, “Edge computing: Vision and challenges,” IEEE Internet Things J., vol. 3, no. 5, pp. 637–646, Oct. 2016.

[10] B. Varghese, M. Villari, O. Rana, P. James, T. Shah, M. Fazio, and R. Ranjan, “Realizing edge marketplaces: Challenges and opportunities,” IEEE Cloud Comput., vol. 5, no. 6, pp. 9–20, Nov. 2018.

[11] A. Ahmed, K. Abu Bakar, M. I. Channa, K. Haseeb, and A. W. Khan, “A survey on trust based detection and isolation of malicious nodes in ad-hoc and sensor networks,” Frontiers Comput. Sci., vol. 9, no. 2, pp. 280–296, Apr. 2015.

[12] J. Guo, J. Ma, X. Li, T. Zhang, and Z. Liu, “A situational awareness trust evolution model for mobile devices in D2D communication,” IEEE Access, vol. 6, pp. 4375–4386, 2018.

[13] J. Yuan, E. Li, C. Q. Kang, F. Y. Chang, and X. Y. Li, “Review of the D2D trusted cooperative mechanism in mobile edge computing,” Information, vol. 10, no. 8, p. 259, Aug. 2019.

[14] S. A. Soleymani, A. H. Abdollah, M. Zareei, M. H. Anisi, C. Vargas-Rosas, M. K. Khan, and S. Goudarzi, “A secure trust model based on fuzzy logic in vehicular ad hoc networks with fog computing,” IEEE Access, vol. 5, pp. 15619–15629, 2017.

[15] O. Ben Abderrahim, M. H. Elhdili, and L. Saidane, “TMCol-SiOT: A trust management system based on communities of interest for the social Internet of Things,” in Proc. 13th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC), Jun. 2017, pp. 747–752.

[16] Z. Gao, C. Xia, Q. Wang, J. Huang, Y. Yang, and L. Rui, “Multi-source feedback based lightweight trust mechanism for edge computing,” in Proc. IEEE Global Commun. Conf. (GLOBECOM), Dec. 2019, pp. 1–6.

[17] F. Jiang and H.-W. Tseng, “Trust model for wireless network security based on the edge computing,” Microsyst. Technol., vol. 25, no. 5, p. 1, May 2019.

[18] X. H. Deng, P. Y. Guan, Z. W. Wan, L. E. Liu, J. Luo, Z. H. Zhao, Y. J. Liu, and H. G. Zhang, “Integrated trust based resource cooperation in edge computing,” in (Chinese), J. Comput. Res. Dev., vol. 55, no. 3, pp. 449–477, Mar. 2018.

[19] X. Huang, R. Yu, J. Kang, and Y. Zhang, “Distributed reputation management for secure and efficient vehicular edge computing and networks,” IEEE Access, vol. 5, pp. 25408–25420, 2017.

[20] Y. He, F. R. Yu, N. Zhao, and H. Yin, “Secure social networks in 5G systems with mobile edge computing, caching, and device-to-device communications,” IEEE Wireless Commun., vol. 25, no. 3, pp. 103–109, Jun. 2018.

[21] Y. Ruan, A. Durresi, and S. Uslu, “Trust assessment for Internet of Things in multi-edge access computing,” in Proc. IEEE 32nd Int. Conf. Adv. Inf. Netw. Appl. (AINA), May 2018, pp. 1155–1161.

[22] Y. B. Ren, X. H. Li, H. Liu, F. Q. Cheng, and J. F. Ma, “Blockchain-based trust management framework for distributed Internet of Things,” in (Chinese), J. Comput. Res. Dev., vol. 55, no. 7, pp. 1462–1478, Jul. 2018.

[23] J. Yuan and X. Li, “A reliable and lightweight trust computing mechanism for IoT edge devices based on multi-source feedback information fusion,” IEEE Access, vol. 6, pp. 23626–23638, 2018.

[24] Z. Gao, W. Zhao, C. Xia, K. Xiao, Z. Mo, Q. Wang, and Y. Yang, “A credible and lightweight multidimensional trust evaluation mechanism for service-oriented IoT edge computing environment,” in Proc. IEEE Int. Congr. Internet Things (ICIOT), Jul. 2019, pp. 156–164.

[25] N. Kammoun, R. Abassi, and S. Guemara, “Towards a new clustering algorithm based on trust management and edge computing for IoT,” in Proc. 15th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC), Jun. 2019, pp. 1570–1575.

[26] X. Ma, Z. Wang, and R. Du, “Trust evaluation model based on multiple service attributes,” J. Netw., vol. 6, no. 6, pp. 842–849, Jun. 2011.

[27] M. S. Khan, M. I. Khan, S.-U.-R. Malik, O. Khalid, M. Azim, and N. Javed, “MATF: A multi-attribute trust framework for MANETs,” EURASIP J. Wireless Commun. Netw., vol. 2016, no. 1, p. 197, Aug. 2016.

[28] X. Ma and X. Li, “Trust evaluation model in edge computing based on integrated trust,” in Proc. Int. Conf. Algorithms, Comput. Artif. Intell. (ACAI), Dec. 2018, pp. 1–6.

[29] S. Nie, “A novel trust model of dynamic optimization based on entropy method in wireless sensor networks,” Cluster Comput., vol. 22, no. S5, pp. 11153–11162, Sep. 2019.

[30] N. Veeraiah and B. T. Krishna, “Trust-aware FuzzyClus-fuzzy Bayesian rule,” Wireless Netw., vol. 25, no. 7, pp. 4021–4035, Jan. 2019.
[31] Y. Du, C. Shi, G.-M. Ma, F.-Y. Chang, B.-P. Xu, Y.-H. Fan, and Z.-T. Sun, “Risk assessment model of power network based on equipment condition and entropy combined weight method,” in *Proc. IEEE Electr. Insul. Conf. (EIC)*, Jun. 2017, pp. 78–82.

[32] B. Li, Y. Zhang, H. Wang, H. Xia, Y. Liu, H. Zhou, and M. Jiang, “Combining various trust factors for E-commerce platforms using analytic hierarchy process,” *J. Softw.*, vol. 9, no. 6, pp. 1604–1611, Jun. 2014.

[33] Y. Wang, T. Uehara, and R. Sasaki, “Fog computing: Issues and challenges in security and forensics,” in *Proc. IEEE 39th Annu. Comput. Softw. Appl. Conf. (COMPSAC)*, Jul. 2015, pp. 53–59.

[34] S. Ganeriwal, L. K. Balzano, and M. Srivastava, “Reputation-based framework for high integrity sensor networks,” *ACM Trans. Sens. Netw.*, vol. 4, no. 3, pp. 1–37, Jan. 2003.

**RUIZHONG DU** was born in 1975. He received the Ph.D. degree in engineering from Wuhan University. He is currently a Professor. His main interests include network security, trusted computing, and network technology.

**KUNQI XU** was born in 1995. Her main research interests include trust evaluation and trusted computing.

**XIAOYAN LIANG** received the master’s degree in computer application and technology from North China Electric Power University, in 2007, and the Ph.D. degree in computer application and technology from Beihang University, in 2016. She is currently a Teacher with Hebei University. Her interests are in network security and semantic analysis.