https://doi.org/10.26599/IJCS.2022.9100016 International Journal of Crowd Science

Product Search Algorithm Based on Improved Ant Colony Optimization in a Distributed Network

Zhishuo Liu[1] ✉, Fang Tian[2], Lida Li[1], Zhuonan Han[1], and Yuqing Li[1]

ABSTRACT

The crowd intelligence-based e-commerce transaction network (CIeTN) is a distributed and unstructured network structure. Smart individuals, such as buyers, sellers, and third-party organizations, can store information in local nodes and connect and share information via moments. The purpose of this study is to design a product search algorithm on the basis of ant colony optimization (ACO) to achieve an efficient and accurate search for the product demand of a node in the network. We introduce the improved ideas of maximum and minimum ants to design a set of heuristic search algorithms on the basis of ACO. To reduce search blindness, additional relevant heuristic factors are selected to define the heuristic calculation equation. The pheromone update mechanism integrating into the product matching factor and forwarding probability is used to design the network search rules among nodes in the search algorithm. Finally, the search algorithm is facilitated by Java language programming and PeerSim software. Experimental results show that the algorithm has significant advantages over the flooding method and the random walk method in terms of search success rate, search time, product matching, search network consumption, and scalability. The search algorithm introduces the idea of improving the maximum and minimum ant colony system and proposes new ideas in the design of heuristic factors in the heuristic equation and the pheromone update strategy. The search algorithm can search for product information effectively.

KEYWORDS

crowd intelligence; distributed network; commodity information search; ant colony optimization

Electronic commerce (e-commerce) has become a mature business model in the context of Internet development. Buyers, sellers, and third-party organizations focus on e-commerce platforms and directly enter platforms forming network structures that use centralized information resource sharing. However, as consumer upgrade demands increase, the limitations of e-commerce platforms in commodity search, commodity recommendation, and incomplete and asymmetric commodity information make consumers dissatisfied. The crowd intelligence-based e-commerce transaction network (CIeTN) is a new type of e-commerce. Individuals, enterprises, and other institutions are connected to a network in the form of agents[1]. Each agent has a different role. All agents have neighbor agents; that is, each agent has its moments. Therefore, the entire information exchange information network of CIeTN is a distributed and unstructured network structure. Buyer and seller nodes store a series of product information related to them in the local node. When the buyer node searches for information about an item, it can not only obtain the item information from the seller node but also gets the purchase record and relevant seller recommendation information from itself so that we can select the item in a diversified and comprehensive way. To achieve the acquisition of commodity information, designing the commodity search algorithm of the entire network is necessary.

Existing research on distributed unstructured network search algorithms can be divided into three types:

(1) Blind search algorithm

The blind search algorithm is the most basic and commonly used search algorithm. It is the basis of other search algorithms, such as flood, random walk, maximum, and iterative incremental search algorithms. It is the first time that a flooding search algorithm has been proposed in the network of Gnutella. The algorithm forwards the search request to all neighbor nodes from the node that generates the demand until the resource is found or when *TTL* is 0. It may traverse all the nodes, which can guarantee the search success rate, but when the network has many nodes, the number of forwarding nodes exponentially increases, occupying resources and taking a long time. Gkantsidis et al.[2], proposed a random walk search algorithm. A node randomly forwards the search request to a neighbor node, which reduces not only the transmission volume but also the search success rate. It may increase the search time. Previous researches[3,4] introduced a modified-BFS search algorithm by improving the flooding algorithm. The algorithm selects the neighbor node that forwards the request according to the calculated probability. Yang and Garcia-Molina[5] proposed the expand ring technology, which periodically queries the depth of the search path during the search process and stops the search when the same search path depth appears. It can reduce the number of unnecessary searches. Cai Kang *et al.*[6] proposed a maximum search algorithm on the basis of the power-law idea. Through the history of a node, select the neighbor node with the largest number of resources for search request forwarding. However, additional resources do not necessarily mean that effective resources exist. A certain degree of blindness can also be observed.

(2) Search algorithm based on node characteristics

The nodes in a network have obvious attribute structure

1 School of Traffic and Transportation, Beijing Jiaotong University, Haidian, Beijing 100089, China
2 Business Administration Division, Seaver College, Pepperdine University, Malibu, CA 90263, USA
Address correspondence to Zhishuo Liu, zhsliu@bjtu.edu.cn

division, which groups the network topology according to a certain level. For example, a network structure is divided into main and secondary nodes. The main nodes are responsible for large numbers of search tasks and information-bearing search algorithms. The main nodes are also connected and secondary, forming a structure in which different node groups are connected. Leibowitz et al.[7] divided nodes into super and ordinary nodes according to their different attributes and capabilities. In topological grouping, supernodes are responsible for all search tasks in the group. Shen et al.[8] proposed the method of index structure and information induction. Ordinary nodes contain all their resource information, whereas supernodes have the resource information of all nodes connected to them, which can realize a fast search. Xiao et al.[9] explored the proportional relationship among different nodes in the network and obtained the optimal number ratio.

(3) Heuristic search algorithm

Heuristic algorithm exploration is based on the above two methods to explore the relationship among nodes. It has been the main research direction in this field in recent years to design effective search request forwarding rules, avoid blind forwarding, and improve search efficiency. Himali and Prasad[10] proposed the search algorithm SPUN. This algorithm records each historical search behavior of a node and calculates the success probability of a neighbor node in searching for different resources in a local node. When the next search is forwarded, the neighbor node with the highest success probability in the first K for forwarding is selected, and a certain TTL value is set as the search termination criterion. This algorithm can reduce request forwarding blindness. However, it is limited in the sense that only the factor of search success rate is considered to select forwarding nodes. Joseph and Hoshiai.[11] proposed a search strategy on the basis of interest domain aggregation. When a resource search request is generated, a node preferentially forwards to a neighbor node with similar interest degrees. The algorithm can improve the search success rate, but the changes in node location consume many resources. The search algorithm based on ant colony optimization (ACO) is also a heuristic search algorithm for distributed unstructured networks. Shojafar et al.[12] set and updated pheromones for different resource node types on the basis of ACO. Selecting forwarding nodes by searching resource types can improve search efficiency. However, the need to clarify resource types makes the algorithm complicated, and it only supports precise search. Li et al.[13] designed an improved resource search algorithm on the basis of node interest and Q-learning. They designed a hotspot resource search mechanism. The algorithm reduces repeated path searches and improves the hit rate compared with other algorithms, resulting in a short response time and reducing redundant information.

Blind search algorithms have shortcomings such as long search time, low search efficiency, unguaranteed search success rate, and high network bandwidth occupancy. ACO can provide a better solution to the distributed problem. Therefore, the purpose of this study is to design a product search algorithm to effectively improve the efficiency of product search. The success rate reduces the broadband network occupation during the search process on the basis of the improved ant colony algorithm for CIeTN. The improvement of the basic ant colony algorithm is based on the improvement strategies proposed by scholars at home and abroad, such as elite strategy ant system, sort-based ant system, maximum–minimum ant system (MMAS), optimal-worst ant system, and ant colony system (ACS).

This research aims at improving the search algorithm design of CIeTN. The algorithm can improve the search efficiency of the entire network, reduce the number of network message transmissions during the search process, reduce the platform operating cost, improve the operating efficiency and user experience, and provide new ideas for the search for commodity information resources in the distributed unstructured network. The rest of this paper is organized as follows. Section 1 introduces the basic definition and algorithm flow of the search algorithm. Section 2 presents the parameter setting of the simulation experiment and the analysis of the experimental results, which are compared with the results of flood search and random walk search algorithms. Section 3 provides the discussion.

1 Experiment

ACO has the characteristics of distribution calculation, positive information feedback, and heuristic search, which is also fully applicable to distributed unstructured network search. The commodity information resource search of CIeTN is similar to the TSP problem. It can be summed up simply as buyer node S demands commodity information resource q in the trading network, and then it sends the search request to its neighbor node. If the neighbor node has the goods information that meets the demand, then it is sent to the S node; otherwise, the search request is forwarded to a neighbor node of the neighbor nodes until q is searched or the entire network is searched.

ACO is suitable for the network search of CIeTN. A node in the network is equivalent to an ant nest, which can release ants to search for commodity resources or carry commodity resources back to demand nodes. Ants select neighbor nodes to forward search requests on the basis of the role of pheromones and enlighten network information. The higher the pheromone concentration, the greater the heuristic factor, and the greater the probability of the path ants searching forward. It can avoid search blindness. The pheromones in the network are updated locally after each search (based on the search effect), resulting in positive feedback. In a certain period T, globally updating such pheromones is suggested to avoid searches that are too early to fall into local optimization.

1.1 Algorithm design

Ant_{search}: when a node receives a commodity information search request for network node search, it generates Ant_{search} to search a neighbor node according to the algorithm forwarding rule. At the same time, a TTL value is assigned. The ant forwards $TTL-1$. When $TTL = 0$ or the required commodity information resource is found to satisfy the termination condition, the ant dies and the search stops.

Ant_{reply}: If the node has a commodity information resource that satisfies the product matching degree, then the node generates Ant_{reply} carrying the commodity information resource and returns the original search request node from the original path, and the ant dies.

Commodity matching degree: it is a standard that measures whether the product information in the node matches the product information of the search demand. This study introduces the idea of a space vector and calculates the cosine similarity of demand commodity information and commodity information in nodes as the matching result of search results. By setting a certain matching degree threshold, if it is exceeded, then it can be identified as a commodity information resource that satisfies the search demand.

The product information is composed of keywords and keyword frequencies, so the $Tf \times IDF$ framework is used to calculate the product information keyword feature weight. Tf is the word frequency, indicating the number of times a keyword appears in the product information. *IDF* is the inverse document frequency factor. The more a word appears in the document collection, the less it can distinguish the difference among documents, and the lower the *IDF* value.

Similar to the words "you" and "me", as they frequently appear in different documents, they cannot be used to distinguish different documents, so the *IDF* values of these words are low. The $Tf \times IDF$ algorithm is used to calculate the keyword weight in the commodity. In Eqs. (1)–(3), $n_{i,j}$ represents the number of times the key_i keyword appears in commodity D_j. $\sum_k n_{k,j}$ indicates the number of occurrences of all keywords in item D_j. N indicates the number of all commodities, and n_i indicates the number of commodities containing key_i.

$$Tf_{i,j} = n_{i,j} / \sum_k n_{k,j} \tag{1}$$

$$IDF_i = \log \frac{N}{n_i} \tag{2}$$

$$\omega_{i,j} = Tf_{i,j} \times IDF_i = \frac{n_{i,j}}{\sum_k n_{k,j}} \times \log \frac{N}{n_i} \tag{3}$$

Calculate product matching degree M_k with the space vector, and represent the query commodity as vector $q\,(m_{i,1}, m_{i,2}, \ldots, m_{i,q})$. Represent the commodity being queried in the node as vector $D_j\,(\omega_{i,1}, \omega_{i,2}, \ldots, \omega_{i,j})$. $m_{i,q}$ is the weight of the keyword key_i in the query commodity, and $\omega_{i,j}$ represents the weight of the keyword key_i in the item being queried in the node, as presented in Eq. (4).

$$M_k = \frac{(q \cdot D_j)}{|q| \times |D_j|} = \left[\sum_{i=1}^{n} (\omega_{i,j} \cdot m_{i,q})\right] \Big/ \left(\sqrt{\sum\nolimits_{i=1}^{n} \omega_{i,j}{}^2} \cdot \sqrt{\sum_{i=1}^{n} m_{i,q}{}^2}\right) \tag{4}$$

Keyword pheromone is taken from node i to neighbor node j searching for ants to update, which is an important factor for calculating the forwarding probability and is represented by $\tau_{ij,key}(t)$. The neighbor node commodity keyword pheromone table on node i records each keyword pheromone concentration on all neighbor nodes. Before the node generates the search ant Ant_{search}, the pheromone concentration in all the neighbor nodes corresponding to the search demand commodity information keyword is calculated. When the neighbor node pheromone concentration is large, the calculated forwarding probability is also large.

The heuristic equation is another important factor in calculating the forwarding probability in the ant colony algorithm, which is represented by $\eta_{ij}(t)$. In this study, three heuristic factors are selected, namely, the internode communication frequency $X(i,j)$, the internode search success rate $Y(i,j)$, and the internode commodity quantity ratio $Z(i,j)$, as presented in Eq. (5).

$$\eta_{ij}(t) = 10 \times [X(i,j) + Y(i,j) + Z(i,j)] \tag{5}$$

The internode communication frequency is represented by $X(i,j)$, which is the ratio of the total number of times node i and neighbor node communicate with each other in the historical communication times of the node i and neighbor node j. When the communication frequency between nodes is high, the possibility of forwarding between nodes in the future is also high. $m_{i,j}$ in Eq. (6) represents the number of historical communication times between node i and neighbor node j. x is a random small positive number, which prevents the ratio of the total number of neighbor nodes of node j to 0. The ratio is $+\infty$. In the actual calculation, x can be 0.000 000 000 01.

$$X(i,j) = m_{i,j} / \sum_{j=1}^{n} m_{i,j} + x \tag{6}$$

The internode search success rate is represented by $Y(i,j)$, which is the proportion of the number of times node i searches for the total number of successful successes of node i in the neighbor node. When the search success rate among nodes is high, the possibility of searching for commodity information resources among nodes in the future is also high. In Eq. (7), $l_{i,j}$ represents the number of historical search successes representing node i and neighbor node j.

$$Y(i,j) = l_{i,j} / \sum_{j=1}^{n} l_{i,j} + x \tag{7}$$

The proportion of the amount of commodity information among nodes is represented by $Z(i,j)$, which is the ratio of the number of commodities of the neighboring node j of node i to the total number of commodities of all the neighboring nodes of node i. When the number of commodity information resources in neighbor node j is large, the possibility of finding the demand commodity information resources is high. In Eq. (8), $o_{i,j}$ represents the number of items of node j, which is the neighbor of node i.

$$Z(i,j) = o_{i,j} / \sum_{j=1}^{n} o_{i,j} + x \tag{8}$$

Forward probability calculation: before the node generates Ant_{search}, it needs to calculate the forwarding probabilities of all neighbor nodes that are represented by $p_{ij}^k(t)$, which is the probability that node i to neighbor node j generates search ants. $p_{ij}^k(t)$ affects the algorithm's transfer rules, as presented in Eqs. (9) and (10). In these equations, $\tau_{ij,key}(t)$ is the keyword pheromone concentration, $\eta_{ij}(t)$ is the heuristic equation, α is the pheromone influence concentration factor, and β is the heuristic equation influence factor. $\tau_{ij,key_r}(t)$ is the pheromone concentration of the keyword in the search product information resource, and r is the number of keyword types in the search product information resource.

$$p_{ij}^k(t) = \begin{cases} \dfrac{[\tau_{ij,key}(t)]^{\alpha} \cdot [\eta_{ij}(t)]^{\beta}}{\sum\limits_{s \in allowed} [\tau_{is,key}(t)]^{\alpha} \cdot [\eta_{is}(t)]^{\beta}}, & \text{if } j \in allowed \\ 0, & \text{else} \end{cases} \tag{9}$$

$$\tau_{ij,key}(t) = \sum_{r=1}^{r} \tau_{ij,key_r}(t) \tag{10}$$

The pheromone update mechanism in this algorithm includes locally and globally updated pheromones.

Local pheromone update rule: when a node generates a reply ant, it takes the commodity information resource and returns it to the demand node. It also updates the keyword pheromone of the node in the return process, as shown in Eqs. (11) and (12). In the formulas, ρ $(0 < \rho < 1)$ is the pheromone volatilization coefficient, and $\triangle\,\tau_{ij,r}(t)$ is a single increment of each keyword pheromone in the neighboring node with the searched commodity information resource. The update of each keyword pheromone is related to the path length of the ant and the product matching degree. The shorter the path taken by the ant, the greater the matching degree of the searched successful product, and the more the pheromone increases. The update of each keyword pheromone is also related to the proportion of the keyword weight of the commodity information resource in the search. In the formula, TTL is the current value, and the larger its current value, the less the path the ant takes. The larger the TTL current value, the less the path the ant walks. M_k is the average value of the matching degree of product returned successfully. ω_r is the keyword weight of the searched product, and r is the number of keyword types of the searched product.

$$\tau_{ij,key_r}(t+n) = (1-\rho) + \tau_{ij,key_r}(t) + \triangle\tau_{ij,r}(t) \tag{11}$$

$$\triangle\tau_{ij,r}(t) = \left(\omega_r / \sum_{i=1}^{r} \omega_i\right) \times (TTL + 10M_k) \tag{12}$$

To prevent the infinite accumulation of pheromones, the pheromone completely affects the transition probability. The full network pheromone is reduced within T, that is, the global pheromone update, see Eq. (13). In the equation, ε is the pheromone volatilization coefficient of period T and $0 < \varepsilon < 1$.

$$\tau_{ij,key_r}(t+T) = (1-\varepsilon) \cdot \tau_{ij,key_r}(t) \tag{13}$$

The tabu table is an ant taboo table, and the nodes that ants walk through in a search are added to it. The ants no longer enter the nodes in the tabu table, and the nodes that other ants can enter are located in the allowed table.

Improvement strategy of ant colony algorithm: in addition to the basic ACO, the search algorithm design of this chapter also adopts the idea of maximal and minimum ACS improvement strategy. The pheromone update in the whole network is between $[\tau_{min}, \tau_{max}]$, which can effectively prevent the influence factor of the heuristic factor from being too large. It can also reduce the influence of the pheromone on the forwarding strategy and improve the convergence speed. Moreover, it can prevent the path pheromone concentration in the late stage of the algorithm from being too large and falling into the local optimum too early.

1.2 Algorithm description

The algorithm divides the product information resource search into two steps: node search and network search.

When a node in a network receives a search request for a product information resource from node S, it first searches and calculates the product matching degree in the local product information resource list and then compares it with the matching degree threshold. If the product information meets the needs, then return to node S.

Otherwise, search for product information resources online. Before a node forwards a search request to a neighbor node, it needs to calculate the ant forwarding probability of the neighbor node and sort this probability, and then determine the neighbor node to forward.

When the required commodity information resource is searched, the node generates feedback ants and updates the pheromone. The search ends. Another scenario is when no suitable product is found but the TTL value is 0, the search ends. The global pheromone is updated again within T.

1.2.1 Node product search algorithm description

The steps of the node product search algorithm are as follows:

(1) Node S in the network generates a search request for commodity information resource$q$, which includes commodity keywords and keyword occurrence frequencies.

(2) Calculate keyword weight q by using Eqs. (1), (2), and (3) in Section 1.1 and generate the keyword weight space vector $q(m_{i,1}, m_{i,2}, \ldots, m_{i,q})$ of q.

(3) Calculate the product matching degree M_k between the product information demand $q(m_{i,1}, m_{i,2}, \ldots, m_{i,q})$ and the product information resource $D_j(\omega_{i,1}, \omega_{i,2}, \ldots, \omega_{i,j})$ in the node by using Eq. (4) in Section 1.1.

(4) Compare M_k with the merchandise matching degree threshold M_0. If $M_k > M_0$, then the product information meets the demand, and the product information resource can be returned to node S.

(5) If M_k is less than M_0 in the nodes searched, set the TTL value and enter the network product information resource search stage.

1.2.2 Online commodity search algorithm description

An information table that stores the keyword pheromones of all neighboring nodes is set on each node. If a local node search fails to find the required commodity information resource, then searching for this resource on neighboring nodes is necessary.

The algorithm sets three heuristic equation factors, including the communication frequency among nodes, the search success rate among nodes, and the proportion of neighbor node products. The forwarding probability of neighbor nodes is calculated by the keyword information concentration and heuristic equations to select forwarding nodes. To give play to the positive algorithm feedback and prevent the infinite accumulation of pheromones, this algorithm adopts the MMAS improvement strategy idea so that the pheromone update in the whole network is between $[\tau_{min}, \tau_{max}]$.

The steps of the online product search algorithm are as follows:

(1) Calculate the forwarding probability. Before the node generates Ant_{search}, calculate the forwarding probability of all neighbor nodes by Eqs. (9) and (10) in Section 1.1 and sort the calculated values. At the same time, give this search a TTL value.

(2) According to the calculation result in Step (1), the node generates K Ant_{search} to forward the search request to the first K neighbor nodes with the largest forwarding probability. At the same time, TTL–1. After Ant_{search} enters the neighbor node, it checks whether the neighbor node is in the table first. If the node is in the tabu table, then Ant_{search} will die; otherwise, Ant_{search} will add this node to the tabu table of this search and start the local node product information resource search. If commodity matching degree M_k is greater than commodity matching degree threshold M_0 in the node, then it generates Ant_{reply}. The Ant_{reply} returns the commodity information resource to the search demand node according to the original path and enters Step (3). If no product matching degree M_k greater than the product matching degree threshold M_0 exists, then the algorithm proceeds to Step (4).

(3) The local pheromone of the node on the way back is updated by Eqs. (11) and (12) in Section 1.1. TTL is the current value. The larger the TTL, the less path the ant takes.

(4) Go back to Steps (1) and (2) until the Ant_{search} finds the commodity information resource needed. Or when TTL = 0, and the required product information resource is still not found, Ant_{search} dies, and the search ends.

(5) In every certain period T, the pheromone in the network is updated globally by Eq. (13) in Section 1.1.

1.2.3 Algorithm flow

The simulation experiment steps of online product search based on the ant colony algorithm are as follows: k is the number of searches, V is the number of search successes, M_k is the product matching degree, M_0 is the product matching degree threshold, and $p_{ij}^{k}(t)$ is the ant forwarding probability.

(1) Randomly generate search demand commodity information

resource q at network random node a, and all nodes in the network currently exist in the allowed table.

(2) Search for the local node commodity information resource at node a, and calculate product matching degree $M_k(k=1)$. The commodity matching degree M_k on the node is compared with the set matching degree threshold M_0. If $M_k \geqslant M_0$, then the target commodity information resource is returned, and M_k is recorded, and the number of successful search times V+1. The number of neighbor nodes communicating with this node and the number of neighbor nodes searching for success must be updated. The algorithm ends. If $M_k < M_0$, then the target commodity is not returned, k remains unchanged. The number of neighbor nodes communicating with the node is updated, the node is added to the tabu, and the process proceeds to Step (3).

(3) Calculate the forwarding probability $p_{ij}^k(t)$ of the neighbor node at this point, take the neighbor node with the forwarding probability before the ranking, and generate Ant_{search} for forwarding, TTL−1. When the ant arrives at the neighbor node, whether the node belongs to the allowed table is determined. If it does, then the node is added to the tabu and proceeds to Step (4); otherwise, Ant_{search} dies.

(4) Query the local commodity information resource and calculate product matching degree M_k. If $M_k \geqslant M_0$, then generate Ant_{reply}; return the commodity information resource to the search request node; and record M_k, $k = k + 1$, and search success frequency V+1. pdate the number of neighbor nodes that communicate with this node and the number of successful neighbor nodes and proceed to Step (5); if $M_k < M_0$, then k remains unchanged, update the number of neighbor nodes that communicate with this node and proceed to Step (6).

(5) In the Ant_{reply} return process, the node is updated according to the algorithm of the commodity keyword pheromone. At this time, Ant_{search} is dead, and the algorithm ends.

(6) Repeat Steps (3) and (4) until the target commodity information is found or the TTL is reduced to 0, the algorithm ends, and Ant_{search} dies.

(7) At T, the commodity keyword pheromone of the nodes in the entire network is globally updated according to the formula; that is, the pheromone is periodically volatilized.

(8) Record and count the search result data every time a search is performed.

2 Simulation

2.1 Evaluation index design

The evaluation index of the search algorithm in this study is divided into two dimensions: the user experience dimension and the network performance dimension.

In the user experience dimension, four evaluation indexes are selected: search success rate, average search time, the average number of search product information resources searched, and average product matching degree.

(1) Search success rate

The search success rate is the ratio of the number of successful searches to the total number of searches in an experiment. The higher the search success rate, the better the search performance of the algorithm in the network. Buyer nodes are more likely to find product information resources than seller nodes.

(2) Average search time

The average search time is the ratio of the total search time to the total number of searches in an experiment. The shorter the average search time, the faster the running speed of the algorithm, and the shorter the waiting time in the search process of buyer nodes.

(3) Average number of search product information resources

The average number of search product information resources is the ratio of the total number of product information resources returned in an experiment to the number of successful searches. This indicator reflects the amount of resource information that a buyer node can return once a search is successful. The larger the number, the greater the user selectivity and reference range for the search results, and the greater the possibility that the user is satisfied with the search results.

(4) Average product match

The average product matching degree is the average of the matching degrees of all returned product information resources in an experiment. The higher the average matching degrees of the products returned by the search algorithm, the better the search matching effect of the algorithm in the network, and the higher the user satisfaction may be.

In terms of network performance dimensions, two evaluation indicators are selected: the average number of message transmission steps and expandability.

(1) Average number of message transmission steps

The average number of message transmission steps is the ratio of the total number of message transmission steps to the total number of searches in an experiment. The message transmission step here refers to each node to which the search request is forwarded. The average number of message transmission steps indicates that the search algorithm occupies network bandwidth resources in each search. The lower the average number of message transmission steps, the less network bandwidth resources are occupied during the algorithm operation.

(2) Scalability

Without changing other variables, gradually increase the number of nodes in the network, and check whether the search success rate and search time of the algorithm are affected. If the search performance is unaffected by the increase of network nodes, then the algorithm scalability is relatively high.

2.2 Experiment design ideas

PeerSim software is used to build a simulation environment for the crowd intelligence-based transaction network. User nodes of a certain size exist in the network, and each node has its unique node identification (ID). The product information is randomly stored on the node according to the data structure designed by the algorithm. Each node is free to join and exit the network and has a certain number of neighbor nodes. The user nodes are divided into buyer and seller nodes according to a certain proportion. Both have different commodity information quantity and attribute characteristics. The search request is randomly generated on the buyer node and searched according to the search algorithm. Finally, the search results are generated.

In the experiment, Java programming is used to implement the ACO-based search algorithm designed in this study. To facilitate the performance of the comparison algorithm, two basic algorithms—flooding and random walk algorithms—are implemented. The written search algorithm is simulated in the network environment of PeerSim. The experimental data are collected through the control interface, and the search results of the three search algorithms are compared. In the simulation experiment, each cycle is experimental, and multiple commodity information resources are searched in one experimental cycle. Each search algorithm first performs multiple cycles and then takes the average of multiple experiments to increase the objectivity and credibility of the experiment.

2.3 Experimental parameter setting

PeerSim simulation software simulates CIeTN. Each node in the network has a unique ID, and each node S circle of friends is connected to five neighbor nodes. The number of summary points in the network is N, which can be set according to the number of experimental nodes. In the simulation experiment, the proportion of buyer nodes in the network is 0.95, that of seller nodes is 0.05, the amount of commodity information owned by buyer nodes is 0–10, and that of commodity information owned by seller nodes is 50–100.

The attributes of the network node, neighbor node information, and pheromone information are all set according to the relevant data structure in the algorithm and are updated by following the algorithm rules. According to the principle of control variables, the static characteristics of commodity information resources and node attributes in the nodes remain unchanged in each experiment.

The parameter setting of the search algorithm based on ACO is shown in Table 1. During the simulation experiment of the ACO-based search algorithm, the parameters are selected according to the following table.

2.4 Analysis

In this study, three search algorithm simulation experiments are carried out, including the flooding, random walk, and ACO algorithms. Each algorithm is subjected to a 10-cycle experiment with 100 searches per cycle experiment. According to each experiment, the evaluation indexes of each algorithm are calculated and then averaged for ten experiments. Finally, the evaluation indexes are obtained. The calculation of the average value can increase the objectivity of the experiment and reduce the error of the experimental data.

According to the experimental data of each algorithm, the algorithm evaluation indexes of search success rate (*SR*), average search time (*AST*), average commodity matching degree (*MD*), average search commodity information resource quantity (*QTY*), and average message transmission step number (*TS*) can be calculated, as presented in Table 2.

(1) Figure 1 compares the *SR*s of the algorithms. The ant colony-based search algorithm has the best effect on the search *SR*, and the average value can reach 76.50%. The random walk search algorithm has a search *SR* of only 20.70%, which is significantly lower than the two other search algorithms. The ant colony-based search algorithm plays an important role in the pheromone and heuristic equations during the search process. Within a certain *TTL*, the search *SR* can reach a high degree.

Table 1 Parameter configuration of search algorithms based on ant colony algorithms.

Parameter	Parameter description	Value
α	Keyword pheromone coefficient	0.5
β	Heuristic coefficient	0.5
ρ	Pheromone volatility	0.1
ε	Global pheromone volatilization coefficient	0.1
M_0	Matching threshold	0.7
o	Number of nodes that generate search ants	3
TTL	Maximum number of searching steps	4
τ_{min}	Minimum pheromone update	5
τ_{max}	Maximum pheromone update	15
T	Global pheromone update cycle	10

Table 2 Calculation results of the evaluation index of search algorithm

Algorithm	*SR* (%)	*AST*	*MD* (%)	*QTY*	*TS*
Flood	58.00	4 618	79.88	19	31.49
Random Walk	20.70	298	80.09	9	2.47
ACO-based	76.50	1 296	83.40	28	15.39

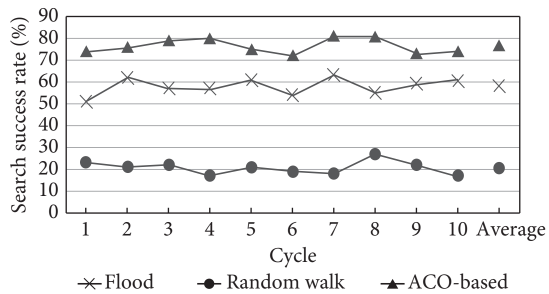

Fig. 1 Search success rate contrast graph.

(2) Figure 2 shows that the average search time of the flooding algorithm per search is significantly higher than that of the two other algorithms, which is also caused by the nature of the algorithm itself. Due to the randomness of its search algorithm, the search time of the random walk algorithm is shortened. In this study, the search time for the designed ACO-based search algorithm is also reduced compared with that for the flooding algorithm. Moreover, the search efficiency of the ACO-based algorithm is about 3.5 times that of the flooding algorithm.

(3) In the flooding and random walk algorithms, the average *MD* of the searched products is the same, about 80%. The ACO-based search algorithm slightly improves the product *MD*, shown in Fig. 3. Mainly due to the algorithm, the attribute characteristics, node behavior characteristics, and the commodity information resource keywords in the nodes are calculated. By searching among nodes with similar features, the obtained commodity information resources match more, and the user may be more satisfied.

(4) Figure 4 indicates that, in a successful search, the number of commodity information resources searched by the ACO-based search algorithm is relatively excellent. In its heuristic calculation, the search for the number of commodity information resources and the neighbor nodes with similar attributes are preferred. Additional results can be found for the user to select.

(5) The search algorithm averages the number of message transmission steps in a search to represent its consumption of broadband resources in the network. The ACO-based search algorithm averages about half the number of message transmission steps in a search, shown in Fig. 5. Therefore, the algorithm designed in this study reduces the consumption of broadband resources in the network by about half compared with the flooding algorithm during the search process.

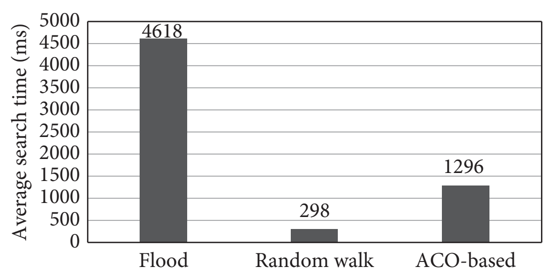

Fig. 2 Average search time contrast chart.

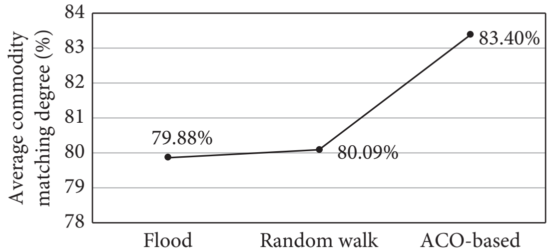

Fig. 3 Average commodity matching degree chart.

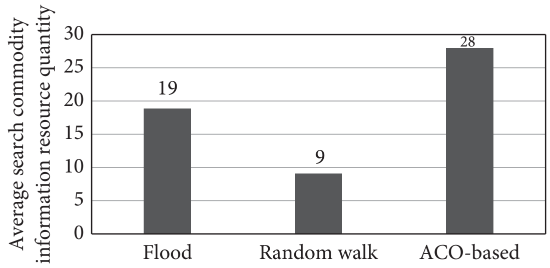

Fig. 4 Contrast map of average search commodity information resources.

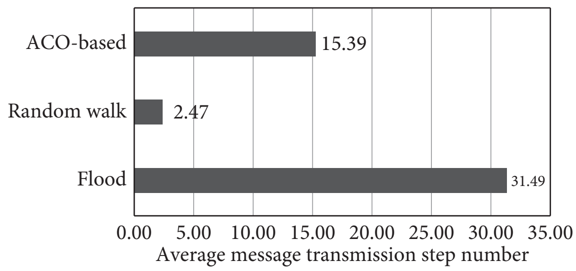

Fig. 5 Average message transfer step contrast graph.

(6) In the scalability test of the search algorithm, the number of network nodes is changed. Other variables are unchanged and carry out simulation experiments on the basis of the ant colony-based search algorithm to check whether the algorithm search *SR* and average search time affect the main evaluation index. The experimental results are shown in Table 3. The algorithm fluctuates within the normal range with the incremental search *SR* and average search time algorithm evaluation index of the node within 50 000 nodes. Therefore, the algorithm has good scalability, and as the network size increases, the search effect of the algorithm will not be affected.

3 Discussion and Conclusion

The experimental results show that the algorithm has some practical value. The heuristic commodity search algorithm based on the improved ant colony algorithm designed in this study combines CIeTN characteristics.

Table 3 Expansibility test results.

Network node	*SR* (%)	*AST*
10 000	76.50	12.96
15 000	75.30	11.48
20 000	76.20	11.98
25 000	74.90	12.56
30 000	72.10	12.47
35 000	73.40	10.87
40 000	75.30	13.55
45 000	71.20	12.46
50 000	73.50	13.01

In terms of transfer probability rule design, relevant heuristic factors are designed on the basis of the behavior characteristics of buyers and sellers in the network. It can balance the function of pheromones effectively according to the historical behavior of nodes and avoid premature stagnation caused by the massive accumulation of pheromones. In terms of pheromone updating rules, not only the traditional pheromone updating path length is considered but also the commodity *MD* is integrated to measure the search effect of CIeTN, so that the pheromone updating strategy can improve the search effect effectively.

Therefore, to some extent, CIeTN can improve the functions of personalized product recommendations and precise consumer demand matching on e-commerce platforms. It can also provide consumers with additional product choices with matched information to improve consumer satisfaction.

Dates

Received: 17 March 2022; Revised: 27 April 2022; Accepted: 28 April 2022

References

[1] Y. Chai, C. Miao, B. Sun, Y. Zheng, and Q. Li, Crowd science and engineering: Concept and research framework, *International Journal of Crowd Science*, vol. 1, no. 1, pp. 2–8, 2017.

[2] C. Gkantsidis, M. Mihail, and A, Saberi, Random walks in peer-to-peer networks: Algorithms and evaluation, *Perform. Eval.*, vol. 63, no. 3, pp. 241–263, 2006.

[3] J. Sugawara and T, Takenaka, Proposal and evaluation of modified-BFS using the number of links in P2P networks, *IEICE Tech. Rep.*, vol. 104, no. 616, pp. 5–8, 2005.

[4] V. Kalogeraki, D. Gunopulos, and D. Zeinalipour-Yazti, A local search mechanism for peer-to-peer networks, in *Proc. 11th Int. Conf. Information and Knowledge Management*, McLean, VA, USA, 2002, pp. 300–307.

[5] B. Yang and H. Garcia-Molina, Improving search in peer-to-peer networks, in *Proc. 22nd Int. Conf. Distributed Systems*, Washington, DC, United States, 2002, pp. 5–14.

[6] K. Cai, H. Tang, S. Ding, and G. Zheng, *P2P Network Principles and Applications*, (in Chinese). Beijing, China: Science Press, 2011.

[7] N. Leibowitz, M. Ripeanu, and A. Wierzbicki, Deconstructing the Kazaa Network, in *Proc. 3rd IEEE Workshop on Internet Applications*, San Jose, CA, USA, 2003, pp. 112–120.

[8] H. T. Shen, Y. Shu, and B, Yu, Efficient semantic-based content search in P2P network, *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 7, pp. 813–826, 2004.

[9] L. Xiao, Z. Zhuang, and Y, Liu, Dynamic layer management in superpeer architectures, *IEEE Trans. Parallel Distrib. Syst.*, vol. 16, no. 11, pp. 1078–1091, 2005.

[10] D. M. R. Himali and S. K. Prasad, SPUN: A P2P probabilistic search algorithm based on successful paths in unstructured networks, in *Proc. 2011 IEEE Int. Symp. Parallel and Distributed Processing Workshops and Phd Forum*, Anchorage, AK, USA, 2011, pp. 1610–1617.

[11] S. Joseph and T, Hoshiai, Decentralized meta-data strategies: Effective peer-to-peer search, *IEICE Trans. Commun.*, vol. 86, no. 6, pp. 1740–1753, 2003.

[12] M. Shojafar, J. H. Abawajy, Z. Delkhah, A. Ahmadi, Z. Pooranian, and A. Abraham, An efficient and distributed file search in unstructured peer-to-peer networks, *Peer-to-Peer Netw. Appl.* , vol. 8, no. 1, pp. 120–136, 2015.

[13] L. F. Li, J. Z. Zhang, P. D. Wang, and P. J. Guo, P2P network search mechanism based on node interest and Q-learning, (in Chinese), *Comput. Sci.* , vol. 47, no. 2, pp. 221–226, 2020.