Unstructured P2P Search Mechanism Based on Ant Colony Optimization

Dongming Tang, Xianliang Lu, Lei Yang

School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China
School of Information Engineering, Southwest University of Science and Technology, Mianyang, China
*1dmtang@hotmail.com, 2xlu@uestc.edu.cn, 3ly2008-stone@163.com

Abstract

Though flooding-based search mechanism has been used extensively in unstructured peer-to-peer network such as Gnutella, the mechanism is not scalable and as a consequence, it consumes a high amount bandwidths and resources. Because of the strong similarities between self-organizing behaviors of ant colonies and self-organization communications in P2P networks, a search algorithm based on Ant Colony Optimization is presented in this paper. By introducing Ant Colony Optimization, each peer maintains routing table, which stores the amount of pheromone corresponding to classification dropped at the link. Based on the pheromone values, a query is flooded to those peers which are most likely to be resources owner. The update of phenomenon depends on the number of documents found and the link cost in query by all ants’ collective cooperation. Simulation results show that, compared with Modified-BFS mechanism, the search algorithm based on Ant Colony Optimization can effectively improve the search performance, and becomes better than the Modified-BFS as the peers optimize their routing tables while using a much smaller number of messages.

Keywords: Ant Colony Optimization, Peer-To-Peer Network, Pheromeon, Information Retrieval

1. Introduction

Over the past few years, Peer-to-peer (P2P) networks, which can offer opportunities for real-time communication, ad-hoc collaboration and information sharing in a large-scale distributed environment, have attracted a great deal of attention from computer science community as well as industrial community. The most distinct characteristic of P2P networks are self organization, without any centralized control and information sharing through direct and symmetric communication between the peers.

File sharing is the most prevalent P2P application in unstructured P2P networks such as Gnutella, in which the key design challenging is efficient technique to search and retrieval of data. Flooding (BFS traversal of the underlying graph) is probably the predominant search technique in unstructured P2P network, and to find a file, a node queries its neighbors, where the query is propagated to all neighbors within a time-to-live (TTL) constraint. Previous researches [1,2] indicated that blind flooding performs well in regular networks but flooding has poor performance, such as causing a large amount of network traffic and uncertain research results.

First proposed by Marco Dorigo in 1992, Ant Colony Optimization (ACO) [3] is a meta-heuristic for combinatorial optimization, and is inspired by the observation of foraging behavior of real ant colonies. While walking from food sources to the nest and vice versa, a moving ant deposits on the ground a chemical substance called pheromone, forming a pheromone trail in this way. When choosing their way, ants can smell pheromone and tend to choose paths marked by strong pheromone concentrations in probability. While an isolated ant moves practically at random, an ant encountering a previously laid trail can detect it and decide to follow it with high probability, thus reinforcing the trail with its own pheromone.

The collective behavior that emerges is a form of autocatalytic process where the more the ants follow a trail, the more attractive that trail becomes to be followed. The process is thus characterized by a positive feedback loop, where the probability with which an ant chooses a path increases with the number of ants that previously chose the same path. It has been shown experimentally that this pheromone trail following behavior can give rise, once employed by a colony of ants, to the emergence of shortest paths. That is, when more paths are available from the nest to a food source, a colony of ants
may be able to exploit the pheromone trails left by the individual ants to discover the shortest path from the nest to the food source and back.

In recent years, researchers were able to transform the models of collective intelligence of ants into useful optimization and control algorithms [4] because of the characteristic in social insect behavior and pheromone-based indirect communications. In ACO [5, 6, 7, 8, 9, 10], a colony of ants is typically modeled as a society of mobile agents and ACO has been applied in many combinatorial optimization problems such as the traveling salesman problem [11], network routing and load-balancing [12], vehicle routing problem [13] and robot path planning [14]. The self-organizing behaviors of ant colonies and self-organization communications in peer-to-peer networks have similarities [15], in which pheromone-based communication is indirect and no requirement to any global knowledge about the network, so ant colony optimization algorithms are suitable for peer-to-peer networks.

In this paper, we focus on search mechanism based on ACO for Gnutella-like decentralized, unstructured P2P systems and we compare the performance of ACO-based search algorithm with that of well-known Modified-BFS. The goal of ACO-based search algorithm is that achieve high efficient in information retrieval.

The balance of the paper is as follows. In Section 2 we briefly review the current related works about research mechanism in unstructured P2P networks and applicability of ACO meta-heuristic to distributed environments. In Section 3 we represent the ACO-based search algorithm in details. In Section 4 we show the experimental results of our algorithm. Finally, we conclude this paper in Section 5.

2. Related Work

Blind flooding technique is the most basic and most direct method to search in unstructured peer-to-peer networks. Because blind flooding consumes a high amount bandwidths and resources, many improved algorithms have been presented. Some algorithms [16] such as iterative deepening, Directed BFS and Local Indices, which were called controlled flooding algorithms, are based on flooding technique and the basic idea is to reduce the number of nodes that process a query. Some algorithms [17, 18] are based on random walk techniques and these more-scalable alternatives have been studied to apply to the general context of unstructured networks to reduce the network traffic. In addition, some methods [19, 20] modify the architecture of P2P systems to promise large performance improvements. They combine the efficiency of the centralized client-server model with the autonomy, load balancing and robustness of distributed search, and also take advantage of heterogeneity of capabilities across peers.

Some algorithms use the principle of locality to increase the efficient of search and resource usage. The Interest-Based Locality algorithm [21] posits that if a peer has a particular piece of content that one is interested in, it is very likely that it will have other items that one is interested in as well. For peer selection, functions that rank the information in the index according to its relevance to the query are used in order to select the node that is most likely to contain content that satisfies the query. The Reputation Learning algorithm [22] creates shortcut links between peers with similar interests and sends queries to the shortcut peers first. Only if there is no result, the query is flooded to the overlay network. The algorithm is based on two premises: one is that peers who would have been able to satisfy previous queries are more likely candidates to answer a current query which is similar and creates shortcut links between peers with similar interests, and another is that peers which share certain resources are more likely to be able to answer each other’s queries because they have at least one common interest. Queries are then sent to the shortcut peers first.

Inspired by the strong similarities between large-scale and dynamic distributed systems and some of the biological environments, Babaoglu et al. [15] propose a conceptual framework that captures several basic biological processes in the form of a family of design patterns and present the applicability of biological processes to distributed environments. Furthermore, they discuss the proliferation-based search algorithms in unstructured overlay networks. The AntHill project [23] presented by Babaoglu et al., which is a Java-based open source framework, aims on design, implementation, and evaluation of P2P applications based on ideas such as multi-agent and evolutionary programming. Caro et al. present AntNet [24], which is an adaptive approach to routing tables learning in packet-switched communication networks. They show that their algorithm based on ant colony optimization outperforms common Internet routing algorithms. Schoonderwoerd et al. [25] apply ant colony optimization to improve load
balancing in telecommunication networks. The results of using the ant-based control is shown to result in fewer call failures than the other methods, compared with those achieved by using fixed shortest-path routes, and also by using an alternative algorithmically-based type of mobile agent previously proposed for use in network management. Michlmayr et al. present the SemAnt [26] which is an ant-based multi-agent system designed for distributed query routing in peer-to-peer network. C.Wu et al. propose AntSearch [27], in which each peer maintains its hit rate of previous queries and records a list of pheromone values of its immediate neighbors, to retrieve more sufficient results and solve the free-riders problem in P2P network.

3. Search Algorithm based on Ant Colony Optimization

The most valuable application of ant algorithms for routing is AntNet [28] which is a novel approach to the adaptive learning of routing table in communication networks. The communication network is mapped on a directed weighted graph with \( N \) nodes and the edges links all the nodes in the graph. All the edges are supposed to be a reliable bit pipes with some cost represented the overhead of bandwidth and transmission delay. Each node manages a routing table and a local traffic statistics. The routing table is organized as in vector-distance algorithm, but the entries are probabilistic value. The structure containing statistics about the local traffic plays the role of local adaptive model for the traffic toward each possible destination. Because AntNet aims to optimize the path to one specific destination node only, so it is not perfect for peer-to-peer search application, where the goal of search mechanism is to find one or more appropriate destinations for a given query keyword.

3.1. Data Structure

In peer-to-peer search application scenario, nodes not only manage and share documents to other nodes but also search destination document based on keyword in the network. Based on AntNet, our ACO-based search algorithm modifies the data structure of node information and also adopts the method of forward ants and backward ants, as well as the strategy for preventing cycles. In search algorithm of peer-to-peer network, the destination node is unknown and content of search packets is variable, so the content of query, which expressed by keyword and syntax, should be considered in the design of routing table.

To describe the data structure, we build two tables called local information table and routing table. Local information table contains the classification data stored, which can support the search quest from other nodes, in local node. Routing table is matrices \( \tau \) of size \( T \times L \), where \( T \) is the number of classification of search keyword and \( L \) is the number of outgoing links for each node. In every peer \( P_i \), routing table stores the amount of pheromone corresponding to classification \( C \) dropped at the link from peer \( P_i \) to peer \( P_n \), where \( P_n \) is the neighbor peer of \( P_i \). Each keyword is represented by a corresponding type of pheromone and the network should independently optimize for each possible keyword by employing multiple pheromone types [30]. A document can be an instance corresponding to one or more concepts, so a query \( Q \) can consist of one or more keywords \( k_1, \ldots, k_n \), which are connected by boolean operator OR. At startup, all table entries are initialized with the same small value \( \tau_{init} = 0.005 \).

3.2. Search Algorithm

Based on the data structure described above, we give the step-by-step description of the ACO-based search algorithm in following six steps.

Step 1 When a search begins, a node firstly searches the local information table by keyword. If the search hits the destination resource, then returns the result to user and record the number of result in variable \( N \). If the number of resource found is less than \( N_{max} \), which \( N_{max} \) is the maximum expectant number of resource, go to step 2. If \( N \) is greater than \( N_{max} \), then search is over.
Step 2 The node generates a forward ant $F_{s \rightarrow d}$, with a startup time $T_{\text{start}}$ and a time-to-live (TTL) parameter $T_{\text{max}}$, to search destination result in the network. Parameter $T_{\text{max}}$ is used to prevent forward ants from running infinitely. When traveling to any node $i$, the forward ant will decide which outgoing link to follow among neighbor peers not already visited. The decision rule is shown in (1).

$$p_j = \begin{cases} \frac{\tau_{kj}}{\sum_{u \in \text{allowed}_i} \tau_{ku}}, & u \in \text{allowed}_i \\ 0, & \text{else} \end{cases}$$

(1)

where $\tau_{kj}$ is the amount of pheromone corresponding to keyword $k$ dropped at path from node $i$ to node $j$. $\text{allowed}_i = \{0, 1, 2, \ldots, n-1\}$ is the allowed peers which forward ant can visited. To increase efficiency of search, we set a parameter $q \in [0, 1)$, which is a threshold value for deciding whether node $j$ should be selected. If $p_j > q$, the forward ants will select $j$ as the next node, and if all $p_j$s less than $q$, then the biggest node $j$ should be selected.

Step 3 In order to increase the efficiency of each search, peer $i$ probes the link congestion by sending ICMP packet to all the target neighbor peers before forward ant set out. Only those neighbor peers who can response the ICMP packet can be selected to be the available neighbor peers. By this means, forward ant avoids entering the paths with link congestion or over subscription.

Step 4 During the traveling, when forward ant reaches a peer, and if the peer has not been visited by any other brother ants, the forward ant will keep memory of its paths and of the traffic condition found. Else if the peer has been visited, the forward ant will select a peer randomly and begin a new search. The identifier of every visited node $k$ and the time elapsed $T_k$ since the launching time to arrive the $k$-th node is pushed in a memory stack. If a forward ant detects a cycle occurring, that is, forward ant is forced to return to an already visited node, the cycle’s nodes are popped from the ant’s stack and all the memory about them is destroyed, the forward ant continues searching. But if the cycle lasted longer than half of the TTL value of forward ant, the forward ant is destroyed.

Step 5 When the keyword is found at peer $d$, the numbers of matching document are recorded in variable $D$ and the time elapse from $T_{\text{start}}$ is recorded in variable $T_d$. Then the forward ant $F_{s \rightarrow d}$ generates a backward ant $B_{d \rightarrow s}$, transfers to it all of its memory. If its TTL value is not decreased to zero, the forward ant is responsible for updating the pheromone trail according to the information gathered by the forward ant by altering the routing table of each visited node. The backward ant $B_{d \rightarrow s}$ travels back hop-by-hop according to the information stored in the memory stack until it arrives at source peer. Similar to [7], the rule of updating pheromone trail is based on (2).

$$\tau_{kj} = \tau_{kj} + \omega \cdot \frac{D}{D_{\max}} + (1 - \omega) \cdot \frac{T_{\text{max}}}{T_d}$$

(2)

where $\omega$ weights the balance of document quantities and cost of time in each query. The change of phenomenon depends on the number of documents found and the link cost in query. So, the more documents found and lower link cost, the more pheromone increased.

Another update rule-pheromone evaporation rule, which is used to all peers in network in a predefined interval to prevent unlimited increment in phenomenon amount, is defined as given in (3).

$$\tau_i = (1 - \rho) \tau_i$$

(3)

where $\rho \in [0, 1]$ is control parameter.

Step 6 When source peer receives all response messages carried by backward ants in $2 \times T_{\text{max}}$, the destination resources are found and the query is terminated. But if there is not any response message
received by source peer, the query is terminated with no result.

4. Performance evaluation

Our simulation is processed by PeerSim [29] and we assume the static network topology and documents distribution. We simulate 1000 peers in the network topology and 150 peers have document resource. The number of document classification is five and respectively corresponding to five keywords and each document is an instance of classification characterized by keyword. Each peer in network can store one or more type document. The related parameters were chosen as follows: $q = 0.6$, $\rho = 0.07$, $T_{\text{max}} = 25$, $D_{\text{max}} = 60$, $\omega = 0.5$. The evaluation metrics used in our simulation include the followings:

(i) Recall rate: the fraction of documents the search mechanism retrieves.

(ii) Number of query messages: defined as the total amount of query messages generated during the searching process.

The recall rate and the number of query messages comparison between ACO-based search algorithm and Modified-BFS (TTL=5) [30] are shown in Figure 1 and Figure 2.

![Figure 1. Recall rate over number of queries.](image1)

![Figure 2. Number of messages over number of queries.](image2)

Figure 1 show that in this situation our mechanism can discover over 90% of the documents over time, while the Modified-BFS stays constant at about 70%. The compare of number of query messages is showed in Figure 2. At first, ACO-based search algorithm must optimize its routing table after several queries, so the number of query of our mechanism is higher than that of Modified-BFS. But the number of messages decreases as the number of queries increases over time while the Modified-BFS mechanism are consistent in our experiment. In conclusion, the simulation results show our mechanism becomes better than the Modified-BFS as the peers optimize their routing tables while using a much smaller number of messages.
5. Conclusion

In this paper we propose and implement a search algorithm based on Ant Colony Optimization in unstructured peer-to-peer network. Based on AntNet, our search mechanism builds the local information table and routing table in each peer and all ants cooperate in creating and updating pheromone stored in routing tables. In every query, query messages are transmitted by the amount of pheromone related to target keyword and if the research hits, positive feedback mechanism will improve the pheromone in right path. The simulation results show the algorithm improves the efficiency of search compared with Modified-BFS mechanism.

6. References

[1] Christos Gkantsidis, Milena Mihail, and Amin Saberi, "Random walks in peer-to-peer networks", in IEEE Infocom, Hong Kong, 2004
[29] peersim.sourceforge.net/, 2011