Journal of Software Engineering and Applications, **2012, 5, 645-652**
http://dx.doi.org/10.4236/jsea.2012.59075 Published Online September 2012 (http://www.SciRP.org/journal/jsea)

Searching Experience Sharing Based on Ant Colony Model

Jung-Sing Jwo*, Chung-Ren Nian

Department of Computer Science, Tunghai University, Taichung, Chinese Taipei.
Email: *jwo@thu.edu.tw

Received May 17th, 2012; revised June 20th, 2012; accepted July 2nd, 2012

ABSTRACT

Search engine is an important tool to all the Internet users. It helps users finding useful contents in the cyberspace. However, searching experiences among different users are difficult to be shared and accumulated. In this paper, a concept called search-trail is proposed. Based on ant colony model, search-trails are created from the searching steps to the target contents. The search-trails built from various users are very similar to the trails generated in an ant colony. The simulations of the proposed solution demonstrate that even in the case of few searching experienced users, the generated search-trails still possess 96.29% similarity to the expected ones in 60 days. It shows that the concept of search-trails can really help users accumulating, sharing and reusing their search experiences.

Keywords: Ant Colony; Internet Search; Pheromone; Search Experience; Search-Trails

1. Introduction

Recently, using the contents and services published in the web to help doing research work is indeed an efficient and useful way. In fact, today's Internet can be considered as the largest knowledge base that has ever existed. In order to identify the required contents in this huge knowledge base, utilizing search engines is the most practical way [1-4].

The process of using a search engine is an interactive process. Web users first submit their keywords and will obtain a set of response URLs; by following these search results users can retrieve the contents of these URLs. This search steps are repeated until either the users are satisfied by what have found or they become frustrated and finally give up. For those people knowing what they are searching, *i.e.*, they know the exact keywords related to the searching topics, current web search solutions are good enough [5,6]. However if the research studies are still in brainstorming stage, "what keywords are the right ones" and "how to identify the next search from the current search" sometimes are very difficult to be answered. Especially, for the users without enough knowledge about the research topic, search engines are still not helpful enough.

From "human" point of view, one of the best ways to tackle the above issues is to reuse someone's experience. Of course, this "someone" should be a knowledgeable guy or a group of knowledgeable ones in the corresponding field. Experience, in fact, is a kind of implicit knowledge which is difficult to be expressed explicitly. So, this raises a new issue: "Can we provide a mechanism such that it can record those knowledgeable guys" search experiences and let others share their experiences later?"

Recording someone's search steps is easy. The difficult part is how to keep "useful" search experiences instead of all the search steps. In order to solve this issue, by mimicking ants' behavior, a new solution called search-trail is proposed. Search-trail treats Internet world as an ant colony [3,7] and each web user is treated as an ant inside the ant colony such that it is able to follow and spread pheromones on the routes to the food sources. In this case, these routes are the search steps approaching to the target contents. As many ants searching the food sources for a while, they can collaborate with each other through pheromone spreading in the area. Some routes will finally become useful search-trails leading to the food sources while others may just disappear. The existing search-trails with strong pheromone can be treated as an associated network of the related keywords and contents. Search-trail map, a visualized interface, is proposed to organize search-trails in a way such that they can be easily reused and shared by either experienced or inexperienced users. Unlike traditional search technology, Search-trail map tries to record how the users search through the web.

In order to see how search-trail map influences the search behaviors, a set of simulations is constructed. In the simulation scenarios, the expected search-trails, *i.e.* the right trails to the food sources, are assumed to be

*Corresponding author.

known. The results of the simulations show that even in the worst case, with 2% experienced, 8% standard, and 90% inexperienced users, the search-trails generated in two months period are 96.29% similar to the expected ones. For all other cases, the search-trails generated in less than 30 days are 100% similar to the expected ones. These simulations show that search-trail map can really help users accumulating and sharing their search experiences. Furthermore, these experiences can be polished day after day just like what ants do in their colony.

This paper is organized as follows. In Section 2, the model of search-trail is introduced. The design of the mechanism is given in Section 3. In order to show the superiority of search-trail map, a set of simulations is discussed in chapter 4. Section 5 is the conclusion remark of this research.

2. Ant Colony Model for Search-Trail

In this section, ant colony model for search-trail is introduced first. Ant colony model is originally inspired by the ants' foraging behavior. In the real world, ants communicate with each other by using an indirect communication called stigmergic [8]. In fact, two ants achieving stigmergic communication is through releasing pheromone. Initially, ants wander randomly until some ants find food. These ants release pheromone on the trails back to their colony. The trails may be found by other ants. Then, other ants may continue wandering randomly or follow the trails they have found. In nature, pheromone evaporates over time. When the pheromone upon a trail evaporates, its attractive strength is reduced. If an ant follows a trail, it may release more pheromone to reinforce the pheromone density. More ants travel through the same trail, more pheromones are spread. Therefore, important trails would remain with higher pheromone density. On the other side, the pheromone evaporation mechanism leads some trails to be discarded. Finally, only good trails remain.

The idea of ant colony model has been applied to many combinational optimization problems, such as traveling salesman problem, assignments and scheduling problems, or routing problems [9-12]. Algorithms based on ant colony model treat ants as separate agents [13-15]. These ants walk around a graph representing the problem to solve. The pheromones released on the edges influence an ant to select its next step. For this kind of algorithms, a mechanism acting as pheromone controller is required. The pheromone controller reduces the density of pheromone level on edges to simulate the pheromone evaporation process. By decreasing the pheromone density, the influences from past experience are reduced and it encourages the exploration of new paths.

The idea of utilizing ant colony model for search-trails is as follows. In a search-trail map, an Internet user is treated as an ant. During its search process, keywords and web pages being visited are recorded as the nodes of a trail. For instance, at beginning a user chooses keyword A to perform search and a set of URLs is returned by the search engine. The user opens a page B from the returned set and then follows a hyperlink available in B to another page C. Then, this user might decide to reformulate the search keyword as D and performs another search. A page E from the new returned set is visited again. The above search process can be modeled as a search-trail as given in **Figure 1(a)**. Note that keywords are in rectangular shapes while pages are in oval shapes. **Figure 1(b)** gives another example of search-trail.

When multiple searches performed, more than one search-trail may appear and a map similar to the trails inside an ant colony can be generated. A web user can choose to follow any existing trail or to create a new one in the map. Pheromone is spread upon the edge of a search-trail when a user is moving from one node to the other thru the edge. If during a period of time an edge of a search-trail has no any access, then a fix amount of pheromone of the edge evaporates.

Based on the above model, users surely will create lots of search-trails after a period of time. Of course, there exist many target contents, *i.e.*, similarly to the food sources in the ant colony, such that no search-trails can reach them. In the map, some search-trails attract more users and become more concrete while others are discarded finally. The detail mechanism for building search-trail map is illustrated in the next section.

3. Search-Trail Map

To develop search-trail map based on ant colony model, pheromone level control is the key issue. Instead of assigning a value to represent pheromone level at the very beginning, the exact time that the pheromone will be completely evaporated is used to represent the initial pheromone level. When an ant travels thru an edge, the

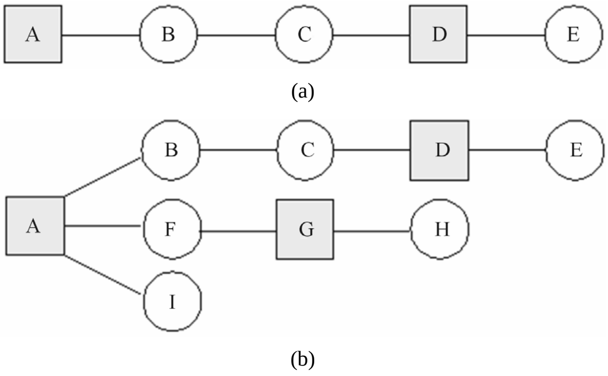


Figure 1. Examples of search-trail model.

pheromone level of the edge is changed to the time by adding a fixed time interval to its previous evaporated time. Since pheromone is spread upon edges in a search-trail map, maintaining each edge's pheromone at its right level is important. When the edges' pheromone levels of a node are all reduced to zero, the node will be removed from the map.

To calculate the remaining pheromone level, it can just subtract the current time from the completely evaporated time. However, the evaporation rate of each edge is not a constant value in a search-trail map. The reason why the evaporation rates are different is that some unpopular but important nodes could be removed from search-trail map after a period of time. Popularity means lots of users are interested in the topic. For example, US President Obama is the topic attracting many searches. Important but not popular is the topic attracting few users. For example, Kenya President Kibaki is important to some web users but definitely is not as popular as President Obama. The search-trails of the two presidents in the search-trail map should maintain readable as long as possible even though the number of the search-trails of President Obama is much larger than the number of President Kibaki. Therefore, the evaporation rate in a search-trail map is adjusted by the popularity. If a node has few out edges to the other nodes, *i.e.*, the degree of the node is small, we say the node is an unpopular node and its out edges' evaporation rate is decreased. On the other hand, a popular node's out edges' evaporation rate is increased. We may wonder under this mechanism the popular nodes should be removed quicker than unpopular nodes. The truth is a popular node's edge usually has more ants traveling thru and therefore the pheromone released on the edge usually is large. In fact, popular nodes are not easy to be removed from a search-trail map.

It is obvious that the complexity of a search-trail map is highly related to the number of the edges in the map. In order to reduce the complexity, limiting a node's degree inside a preferred bound is reasonable since too many edges are not helpful for searching.

Before introducing the solution in detail, we first summarize all variables in **Table 1**.

In order to guarantee a newly created edge can be noticed by other users initially, the *initial_pheromone_value* usually should be given a larger value. On the other hand, for existing edges, if they can lead to some important nodes, their pheromone values are accumulated by passing-by users and therefore the *pheromone_spread_value* is not necessary as big as the value of *initial_pheromone_value.*

The data structures of Node, Edge, and Ant are given in **Figures 2-4** respectively.

Figure 5 is the major algorithm for search-trail solution. In this algorithm, when an *ant* moves from its current node to another node named *nextNode*, the method *Ant_Movement* (*ant*, *nextNode*) is invoked. The algorithm starts by assigning the *ant*'*s* current location as *firstNode*, then it checks whether edge *e* = (*firstNode*, *nextNode*) has already existed in the search-trail map. If it does, then the number of ants traveling thru *e* during the current time interval is increased by one. If *e* is not existed in the search-trail map, then *e* is created by executing the *Edge_Creation* (*firstNode*, *nextNode*) method and *e* is added into the search-trail map.

After a short period of time *pheromone_update_time_interval*, a search-trail map needs to update the pheromone level of each edge. The algorithm is listed in **Figure 6**. In this algorithm, each edge's *evaporation_rate* is

Table 1. Variable definitions.

Variables	Description
evaporation_rate	This variable is an edge variable. It indicates pheromone evaporation rate of a given edge.
edge_count	This variable is a node variable. It indicates the current number of out edges of a given node.
preferred_edge_count	This variable is a global variable. It indicates the maximum number of out edges of any node can exist in a search-trail map.
pheromone_update_time_interval	This variable is a global variable. It indicates the time interval to modify an edge's pheromone in a search-trail map.
completely_evaporated_time	This variable is an edge variable. It indicates the time that an edge's pheromone will be completely evaporated.
current_time	This variable is a global variable. It indicates current time.
initial_pheromone_value	This variable is a global variable. It indicates the initial pheromone value spread on a new created edge.
pheromone_spread_value	This variable is a global variable. It indicates the amount of pheromone spread on an edge when a user travels thru it.
number_of_ants	This variable is an edge variable. It indicates the number of users that travelled through the edge during a *pheromone_update_time_interval.*
Current	This variable is an ant variable. It indicates the current position of the ant.

Node type
edge_count

Figure 2. Data structure for node.

Edge type
firstNode
nextNode
evaporation_rate
number_of_ants
completely_evaporated_time

Figure 3. Data structure for edge.

Ant type
Current

Figure 4. Data structure for ant.

```
Ant_Movement (Ant ant, Node nextNode){
    define Node firstNode = Ant.current;
    if   (firstNode, nextNode) is an existing
         edge in search-trail map {
            let e = (firstNode, nextNode);
            Ant_count (e);
         } else {
            e = Edge_Creation (firstNode,
                nextNode);
            add e to search-trail map;
    }
    ant.current  ←  nextNode;
}

Edge_Creation (Node firstNode, Node nextNode){
    if nextNode doesn't exist, create a new node
      and assign the new node to nextNode;
      define Edge e = (firstNode, nextNode);
    e. completely_evaporated_time  ←
      current_time + initial_pheromone_value;
    e.evaporation_rate  ←  1;
    return e;
}

Ant_Count (Edge e){
      e. number_of_ants  ←
          e. number_of_ants + 1;
}
```

Figure 5. Ant movement algorithm.

```
ST-Map_Update (){
    for each Edge e=(firstNode, nextNode) in ST-Map{
      e.evaporation_rate ← e.evaporation_rate ×
        $\frac{preferred\_edge\_count}{e.firstNode.edge\_count}$;
      e.completely_evaporated_time ← current_time +
        $\frac{e.completely\_evaporated\_time - current\_time}{e.evaporation\_rate}$ +
        $pheromone\_spread\_time \times e.number\_of\_ants$;
      If e.completely_evaporated_time < current_time {
        remove e from ST-Map;
        e.firstNode.edge_count =
           e. firstNode.edge_count – 1;
      }
      If e.firstNode.edge_count <= 0{
        remove e.firstNode from ST-Map;
      }
    }
}
```

Figure 6. Search-trail map pheromone update algorithm.

updated according to its popularity. When *evaporation_rate* is larger than one, it is said to be popular. Otherwise, it is unpopular. The value of *completely_evaporated_time* is updated according to the formula given in the **Figure 6**. After updating *completely_evaporated_time*, it verifies whether the edge is already evaporated by comparing it with *current_time*. If *completely_evaporated_time* is smaller than the *current_time*, which means the edge is evaporated, the edge should be removed from the search-trail map. If an edge is removed from the map, its *firstNode*'s *edge_count* should also be reduced by one. When a node's *edge_count* becomes zero, it is removed from the map.

4. Search-Trail Simulation

In order to see the impact on the web search when adopting search-trail, an experiment is proposed. The input to this experiment is as follows. There are 900 nodes and 3417 edges having been visited and gone thru by many ants after a long enough period of time. Inside this colony, there are totally 83 keywords and web contents considered as "good" targets. **Figure 7** is the visualization map of the input. The trails with dark grey color are the *expected good search-trails* to be generated after simulation.

The simulation mechanism is given in **Figure 8**. It starts by generating the above input map named "*Reference Map*". Then, an Ant Dispatcher creates ants in every *dispatch_ant_time_interval* time period. As many ants join searching, some search-trails on the map become strengthened. Meanwhile, according to the algorithm stated in the previous section, the Pheromone Controller is developed to control the pheromone level on the edges of the simulated map. In the simulation, there are three different categories of ants in the colony, namely, experienced users, standard users, and inexperienced users. Each of these ants, its search experience is decided by *experience_level* which indicates the probability of choosing good nodes by its own. The average time for an ant staying on a node is defined by *sleep_time*. The probability of following existing search-trails is stored in

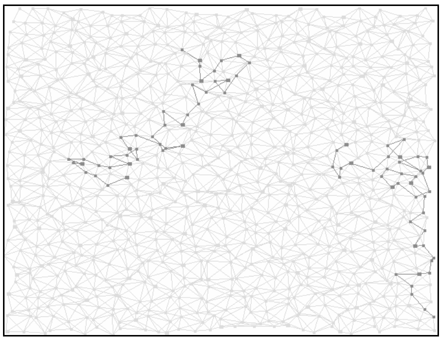


Figure 7. Scenario for the experiments.

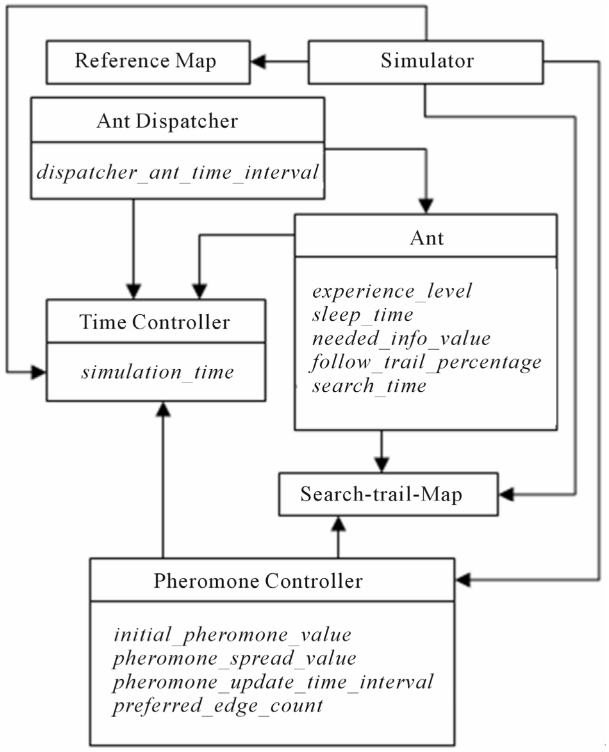


Figure 8. Class diagram for simulation.

follow_trail_percentage. The ant will finish its search either the time it spends for search is more than *search_time* or the good nodes it has found is more than *satisfied_node_number*. Finally, the whole simulation is driven by Time Controller. In the Time Controller, *simulation_time* represents the duration of simulated days. **Table 2** lists all the key attributes of the classes.

Tables 3 and **4** contains the attribute values used in the simulations done in this section. In order to see how good the search-trails generated by the ants after 30 days in each simulation, a similarity formula to show the difference between the expected search-trails and the generated search-trails is given in the following:

$$\text{Similarity} = \frac{\text{edges in }(\text{geneated search trails} \cap \text{expected search trails})}{\text{edges in }(\text{geneated search trails} \cup \text{expected search trails})}$$

In the formula, the numerator represents the generated search-trails that are expected and the denominator represents the expected search-trails plus the wrong ones generated by the ants. If all the search-trails generated by the ants are expected, the value of the similarity will be 100%.

Based on the above introduction, four simulations are performed. For Simulation 1, we assume 8% experienced, 32% standard, and 60% inexperienced users appearing in the colony, respectively. For Simulation 2%, 6%, 24%, and 70% of the three different category ants are assumed. For Simulation 3, 4%, 16%, and 80% are assumed. For Simulation 4, 2%, 8%, and 90% are assumed. **Figures 9-12** are the results of the above four simulations respectively.

In the above four simulations, each time we perform a new simulation, the amount of inexperienced user is increased while the amount of standard and experienced user is decreased relatively. The search-trails generated after day 1 are really random for all the four cases. However, when more and more ants join into the search, with the pheromone spread and evaporated, the search-trails generated in the four cases are all gradually converged to the expected search-trails. In fact, the first two cases generate the exact expected search-trails in 30 days.

Table 2. Component attributes.

Component	Parameter	Description
Time Controller	*Simulation_time*	It indicates the period of simulated time.
Ant Dispatcher	*dispatch_ant_time_interval*	It indicates the frequency of an ant's creation.
Ant	*Experience_level*	It indicates the experience level of the ant.
	sleep_time	It indicates the time for the ant staying in a node.
	satisfied_ node_value	It indicates the number of the nodes the ant visited and then it will finish the search.
	follow_trail_percentage	It indicates the probability of the ant choosing existing search-trails.
	search_time	It indicates the time period the ant stay in searching. It is a random variable in normal distribution.

Table 3. Key component parameters.

Component	Parameter	Value
Time Controller	*simulation_time*	30 days
Pheromone Controller	*initial_pheromone_value*	3 days
	pheromone_spread_value	1 hour
	pheromone_update_time_interval	3 hours
	preferred_edge_count	3
Ant Dispatcher	*dispatch_ant_time_interval*	3 min.

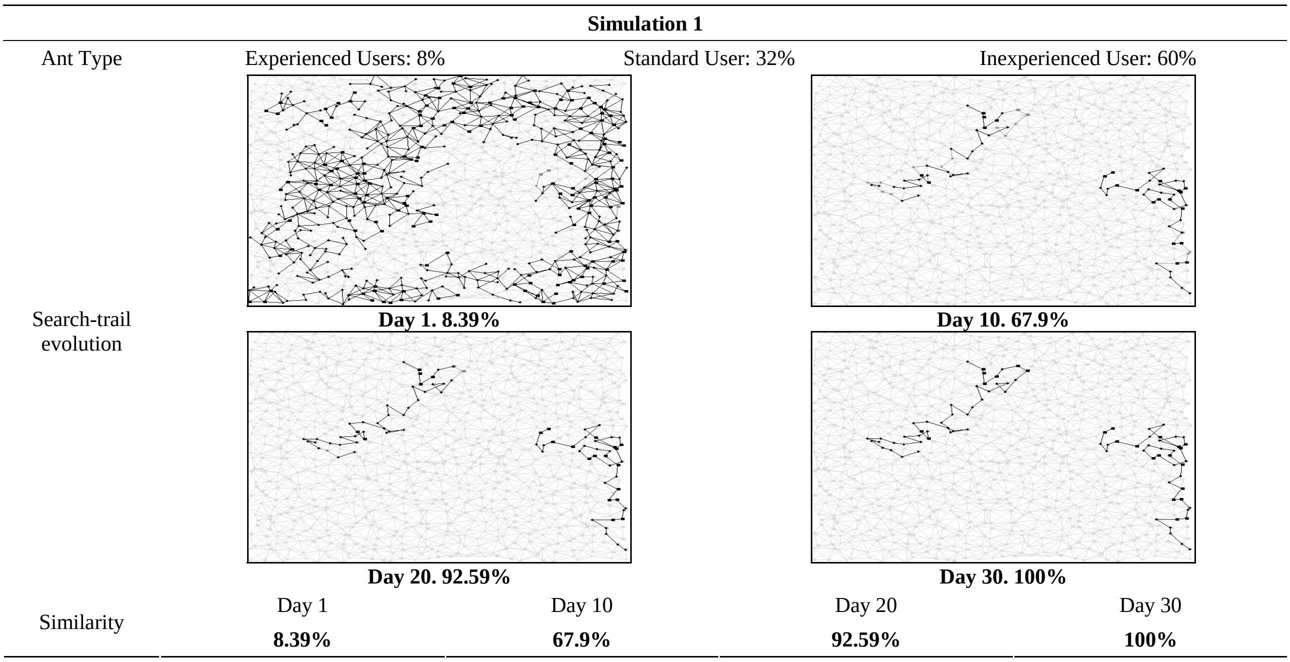


Figure 9. Simulation 1.

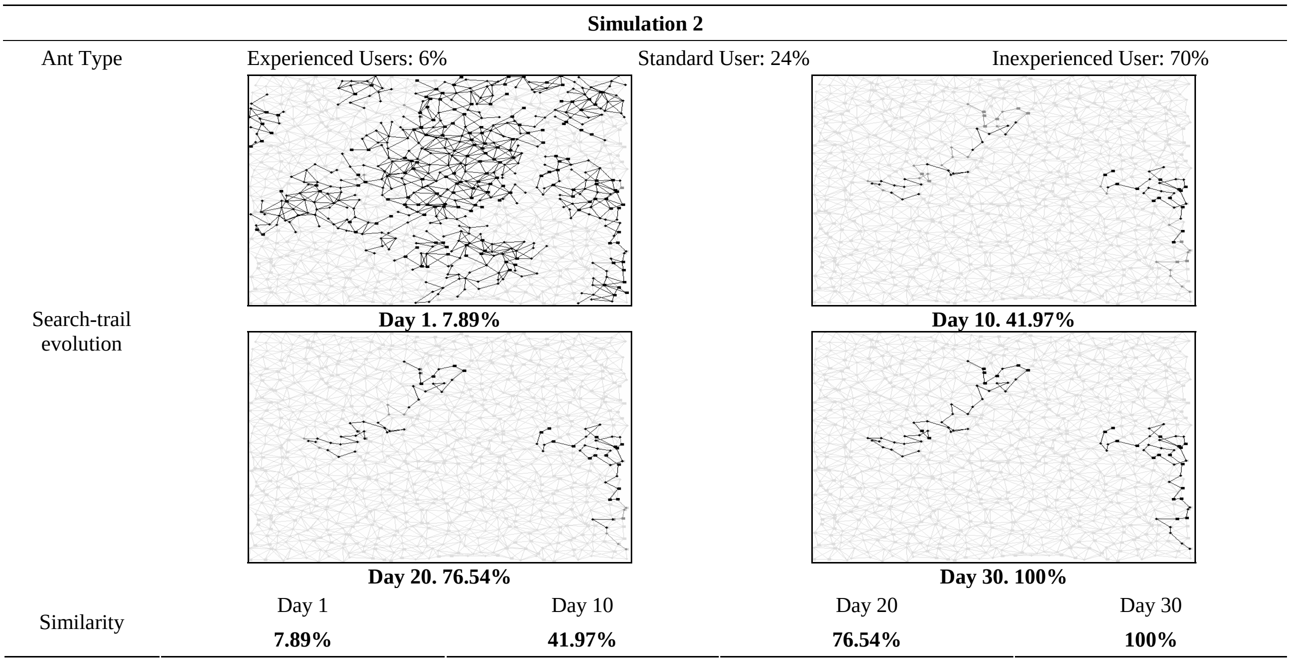


Figure 10. Simulation 2.

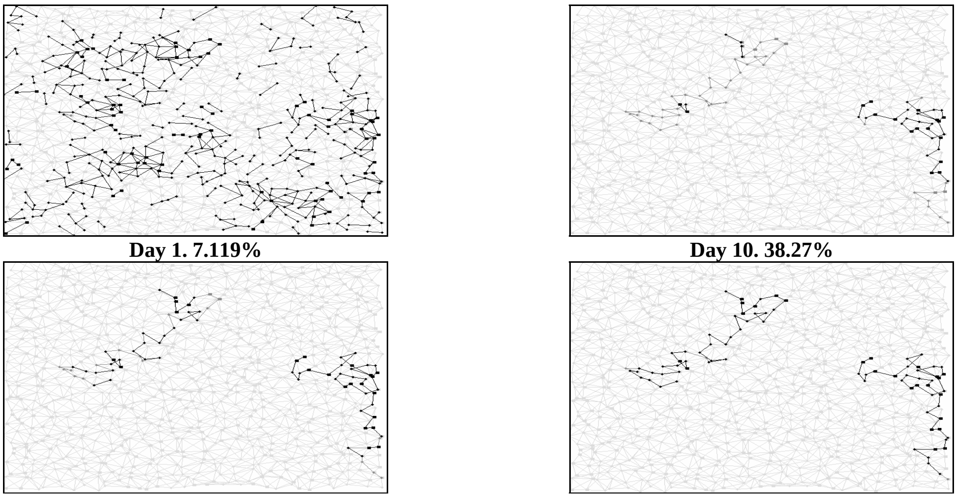
Table 4. Ants parameters.

Parameter	Experienced Users	Standard Users	Inexperienced Users
experience_level	75%	50%	25%
sleep_time	$\mu = 30$ sec $\sigma^2 = 5$	$\mu = 30$ sec $\sigma^2 = 5$	$\mu = 30$ sec $\sigma^2 = 5$
Satisified_node_value	$\mu = 20$ $\sigma^2 = 10$	$\mu = 20$ $\sigma^2 = 10$	$\mu = 20$ $\sigma^2 = 10$
follow_trail_percentage	50%	50%	50%
search_time	$\mu = 25$ min $\sigma^2 = 5$	$\mu = 25$ min $\sigma^2 = 5$	$\mu = 25$ min $\sigma^2 = 5$

Simulation 3

Ant Type: Experienced Users: 4% Standard User: 16% Inexperienced User: 80%

search-trail evolution



Day 1. 7.119%

Day 10. 38.27%

Day 20. 66.66%

Day 30. 91.35%

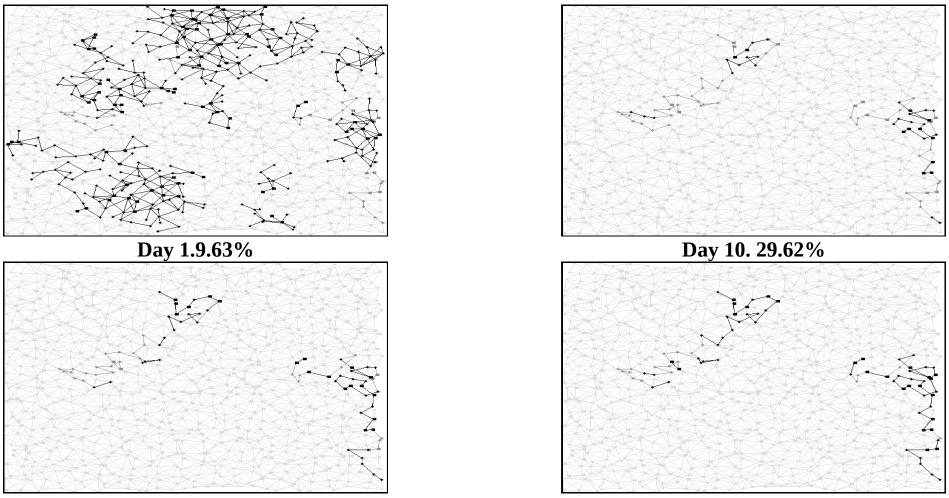
	Day 1	Day 10	Day 20	Day 30
Similarity	**7.11%**	**38.27%**	**66.66%**	**91.35%**

Figure 11. Simulation 3.

Simulation 4

Ant Type: Experienced Users: 2% Standard User: 8% Inexperienced User: 90%

search-trail evolution



Day 1.9.63%

Day 10. 29.62%

Day 20. 51.85%

Day 30. 72.83%

	Day 1	Day 10	Day 20	Day 30
Similarity	**9.63%**	**29.62%**	**51.85%**	**72.83%**

Figure 12. Simulation 4.

For the forth simulation, if the simulation period is extended to 60 days, the generated search-trail similarity can reach 96.29%.

5. Conclusions

In this research a concept called search-trail is proposed. Instead of the search targets, search-trail focuses more on the search process. Based on ant colony model, search-trails are created from the searching steps to the target contents. The search-trails built from various users are very similar to the trails generated in an ant colony. In order to see the impact when adopting the proposed solution in a search community, four different simulations are performed. The results show that search-trails really can help users accumulating and sharing search experiences. Even in the case of very few experienced users, the generated search-trails still possess 96.29% similarity to the expected ones in 60 days.

The future work of this research is to integrate search-trail into the current available search engines. In order to achieve this goal, the complexity of the solution is required to be further polished.

REFERENCES

[1] P. Cimiano and S. Staab, "Learning by Googling," *SIGKDD Explorations*, Vol. 6, No. 2, 2004, pp. 24-33. doi:10.1145/1046456.1046460

[2] S. Lawrence and G. C. Lee, "Accessibility of Information on the Web," *Nature*, Vol. 400, 1999, pp. 107-109. doi:10.1038/21987

[3] S. Lawrence and G. C. Lee, "Searching the Web: General and Scientific Information Access," *IEEE Communications Magazine*, Vol. 37, No. 1, 1999, pp. 116-122. doi:10.1109/35.739314

[4] J. Pokorny, "Web Searching and Information Retrieval," *Computing in Science & Engineering*, Vol. 6, No. 4, 2004, pp. 43-48. doi:10.1109/MCSE.2004.24

[5] A. Spink, R. Building, D. Wolfram and T. Saracevic, "Searching the Web: The Public and Their Queries," *Journal of the American Society for Information Science and Technology*, Vol. 52, 2001, pp. 226-234. doi:10.1002/1097-4571(2000)9999:9999<::AID-ASI1591>3.0.CO;2-R

[6] A. Spink, B. J. Jansen, C. Blakely and S. Koshman, "Overlap among Major Web Search Engines," 3*rd International Conference on Information Technology*: *New Generation*, Las Vegas, 10-12 April 2006, pp. 370-374.

[7] S. Goss, R. Beckers, J. L. Deneubourg, S. Aron and J. M. Pasteels, "How Trail Laying and Trail Following Can Solve Foraging Problems for Ant Colonies," *Behavioral Mechanism of Food Selection*, *NATO ASI Series*, Vol. G20, 1990, pp. 661-678. doi:10.1007/978-3-642-75118-9_32

[8] J.-L. Deneubourg, S. Aron, S. Goss and J.-M. Pasteels, "The Self-Organizing Exploratory Pattern of the Argentine Ant," *Journal of Insect Behavior*, Vol. 3, No. 2, 1990, pp. 159-168. doi:10.1007/BF01417909

[9] C. Blum and M. Sampels, "An Ant Colony Optimization Slgorithm for Shop Scheduling Problems," *Journal of Mathematical Modelling and Algorithms*, Vol. 3, No. 2, 004, pp. 285-308.

[10] M. Dorigo and L. M. Gambardella, "Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem," *IEEE Transactions on Evolutionary Computation*, Vol. 1, No. 1, 1997, pp. 53-66. doi:10.1109/4235.585892

[11] O. H. Hussein, T. N. Saadawi and M. J. Lee, "Probability Routing Algorithm for Mobile Ad Hoc Networks Resources Management," *IEEE Journal on Selected Areas in Communications*, Vol. 23, No. 12, 2005, pp. 2248-2259. doi:10.1109/JSAC.2005.857205

[12] K. M. Sim and W. H. Sun, "Ant Colony Optimization for Routing and Load-Balancing," *IEEE Transactions on Systems*, *Man and Cybernetics*, Vol. 33, No. 5, 2003, pp. 560-572. doi:10.1109/TSMCA.2003.817391

[13] M. Dorigo, G. D. Caro and L. M. Gambardella, "Ant Algorithms for Discrete Optimization," *Artificial Life*, Vol. 5, No. 2, 1999, pp. 137-172. doi:10.1162/106454699568728

[14] J. F. A. Traniello, "Colony Specificity in the Trail Pheromone of an Ant," *Naturwissenschaften*, Vol. 67, No. 7, 1980, pp. 361-362. doi:10.1007/BF01106597

[15] S. Zheng, G. Zhang and Z. Zhou, "Ant Colony Optimization Based on Pheromone Trail Centralization," *Proceedings of the* 6*th World Congress on Intelligent Control and Automation*, Vol. 1, 2006, pp. 3349-3352.