

# Spam Host Detection Using Ant Colony Optimization

Arnon Rungsawang, Apichat Taweessiriwate  
and Bundit Manaskasemsak

**Abstract** Inappropriate effort of web manipulation or spamming in order to boost up a web page into the first rank of a search result is an important problem, and affects the efficiency of a search engine. This article presents a spam host detection approach. We exploit both content and link features extracting from hosts to train a learning model based on ant colony optimization algorithm. Experiments on the WEBSPAM-UK2006 dataset show that the proposed method provides higher precision in detecting spam than the baseline C.45 and SVM.

**Keywords** Spam host detection • Ant colony optimization algorithm • Content and link features • Search engine

## 1 Introduction

Search Engine has been developed and used as a tool to locate web information and resources. For a given query, the ranking result on the first page of a famous search engine is highly valuable to commercial web sites. Current competitive business then gives birth to aggressive attempts from web engineers to boost the

---

A. Rungsawang (✉) · A. Taweessiriwate · B. Manaskasemsak  
Massive Information and Knowledge Engineering Department of Computer Engineering,  
Kasetsart University, Bangkok 10900, Thailand  
e-mail: arnon@mikelab.net

A. Taweessiriwate  
e-mail: ball@mikelab.net

B. Manaskasemsak  
e-mail: un@mikelab.net

ranking of web pages in search results to increase the return of investment (ROI). Manipulating search engine ranking methods to obtain a higher than deserved rank of a web page is called search engine (or web) spam [1]. Besides degrading the quality of search results, the large number of pages explicitly created for spamming also increases the cost of crawling, and inflates both index and storage with many useless pages.

As described by Gyöngyi and Garcia-Molina in [1], there are many varieties of spamming techniques. Often, most of them exploit the weakness of the search engine's ranking algorithm, such as inserting a large number of words that are unrelated to the main content of the page (i.e., content spam), or creating a link farm to spoil the link-based ranking results (i.e., link spam). Many researchers have concentrated on combating spam. For example, Gyöngyi et al. [2] propose an idea to propagate trust from good sites to demote spam, while Wu and Davison [3] expand from a seed set of spam pages to the neighbors to find more suspicious pages in the web graph. Dai et al. [4] exploit the historical content information of web pages to improve spam classification, while Chung et al. [5] propose to use time series to study the link farm evolution. Martinez-Romo and Araujo [6] apply a language model approach to improve web spam identification.

In this paper, we propose to apply the ant colony optimization algorithm [7, 8] in detecting spam host problem. Both content and link based features extracted from normal and spam hosts have been used to train the classification model in order to discover a list of classification rules. From the experiments with the WEBSPPAM-UK2006 [9], the results show that rules generated from ant colony optimization learning model can classify spam hosts more precise than the base-line decision tree (C4.5 algorithm) and support vector machine (SVM) models, that have been explored by many researchers [10–12].

## 2 Related Work and Basic Concept

### 2.1 *Web Spam Detection Using Machine Learning Techniques*

Web spam detection became a known topic to academic discourse since the Davison's paper on using machine learning techniques to identify link spam [13], and was further reasserted by Henzinger et al. [14] as one of the most challenges to commercial search engines. Web spam detection can be seen as a binary classification problem; a page or host will be predicted as spam or not spam.

Fetterly et al. [15] observe the distribution of statistical properties of web pages and found that they can be used to identify spam. In addition to content properties of the web pages or hosts, link data is also very helpful. Becchetti et al. [10] exploit the link features, e.g., the number of in- and out-degree, PageRank [16], and TrustRank [2], to build a spam classifier. Following the work in [10, 12], Castillo et al. [11]

extract link features from the web graph and host graph, and content features from individual pages, and use the simple decision tree C4.5 to build the classifier. Recently, Dai et al. [4] extract temporal features from the Internet Archive's Wayback Machine [17] and use them to train a cascade classifier built from several SVM<sup>light</sup> and a logistic regression implemented in WEKA [18].

## 2.2 Basic Concept of Ant Colony Optimization

Naturally, distinct kind of creatures behaves differently in their everyday life. In a colony of social ants, each ant usually has its own duty and performs its own tasks independently from other members of the colony. However, tasks done by different ants are usually related to each other in such a way that the colony, as a whole, is capable of solving complex problems through cooperation [8, 19]. For example, for survival-related problems such as selecting the shortest walking path, finding and storing food, which require sophisticated planning, are solved by ant colony without any kind of supervisor. The extensive study from ethologists reveals that ants communicate with one another by means of pheromone trails to exchange information about which path should be followed. As ants move, a certain amount of pheromone is dropped to make the path with the trail of this substance. Ants tend to converge to the shortest trail (or path), since they can make more trips, and hence deliver more food to their colony. The more ants follow a given trail, the more attractive this trail becomes to be followed by other ants. This process can be described as a positive feedback loop, in which the probability that an ant chooses a path is proportional to the number of ants that has already passed through that path [7, 8].

Researchers try to simulate the natural behavior of ants, including mechanisms of cooperation, and devise ant colony optimization (ACO) algorithms based on such an idea to solve the real world complex problems, such as the travelling salesman problem [20], data mining [19]. ACO algorithms solve a problem based on the following concept:

- Each path followed by an ant is associated with a candidate solution for a given problem.
- When an ant follows a path, it drops varying amount of pheromone on that path in proportion with the quality of the corresponding candidate solution for the target problem.
- Path with a larger amount of pheromone will have a greater probability to be chosen to follow by other ants.

In solving an optimization problem with ACO, we have to choose three following functions appropriately to help the algorithm to get faster and better solution. The first one is a problem-dependent heuristic function ( $\eta$ ) which measures the quality of items (i.e., attribute-value pairs) that can be added to the current partial solution (i.e., rule). The second one is a rule for pheromone updating ( $\tau$ ) which specifies how to modify the pheromone trail. The last one is a

probabilistic transition rule ( $P$ ) based on the value of the heuristic function and on the contents of the pheromone trail that is used to iteratively construct the solution.

### 3 Spam Detection Based on Ant Colony Optimization Algorithm

#### 3.1 Graph Representation

In a learning process based on the ACO algorithm, problems are often modeled as a graph. Thus, we let  $\{A_1, A_2, \dots, A_m\}$  represent a set of  $m$  features, i.e., both content and link features, extracted from hosts. If we denote  $\{a_{i1}, a_{i2}, \dots, a_{ini}\}$  to a set of  $n_i$  possible values belonged to a feature  $A_i$ . Therefore, we can construct a graph  $G = (V, E)$  including a set of nodes  $V = \{A_1, A_2, \dots, A_m\} \cup \{S\}$  and a set of edges  $E = V^2$ , where  $S$  is a virtual node set to a starting point. This graph can be illustrated in Fig. 1.

#### 3.2 Methodology

Consider the graph in Fig. 1, when we assign artificial ants to start walking from node  $S$ , behavior of those ants will decide to choose a path to walk in each step, from one node to others, using some probabilistic transition function calculated based on the value of a heuristic function and pheromone information value. The following probabilistic transition  $P_{ij}$  is denoted a probability value for an ant to walk from any current node to node  $a_{ij}$ :

$$P_{ij} = \frac{\eta_{ij}\tau_{ij}(t)}{\sum_{i=1}^m x_i \cdot \sum_{j=1}^{n_i} (\eta_{ij}\tau_{ij}(t))}, \quad (1)$$

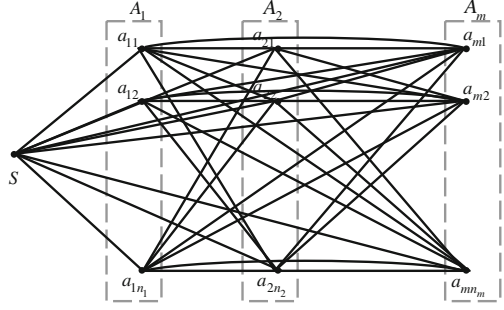
where  $\eta_{ij}$  denotes a heuristic function,  $\tau_{ij}(t)$  denotes a pheromone information value obtained at iteration time  $t$ , and

$$x_i = \begin{cases} 1 & \text{If the node } a_{i*} \text{ has never been passed by that ant,} \\ 0 & \text{Otherwise.} \end{cases} \quad (2)$$

In this paper, we use an open source software package called GUIAnt-Miner [21] which provides an implementation of the ACO algorithm [19] used for classification problems in data mining. The default heuristic function with the value of disorder (i.e., the entropy function) between nodes is defined by:

$$\eta_{ij} = \frac{\log_2 k - H(W|A_i = a_{ij})}{\sum_{i=1}^m x_i \cdot \sum_{j=1}^{n_i} (\log_2 k - H(W|A_i = a_{ij}))}, \quad (3)$$

**Fig. 1** The problem represented as a graph



where

$$H(W|A_i = a_{ij}) = - \sum_{w=1}^k (P(w|A_i = a_{ij}) \cdot \log_2 P(w|A_i = a_{ij})). \quad (4)$$

Here, we define  $W$  as a set of target classes and  $k$  as the number of classes (i.e.,  $|W|$ ), so that  $W = \{spam, normal\}$  and  $k = 2$  in this case.  $P(w|A_i = a_{ij})$  is the probability of class  $w$  given  $A_i = a_{ij}$ . Consequently, the range of a value obtained from Eq. 4 is  $(0, \log_2 k)$ .

Since the ACO algorithm iteratively finds the optimal solution, the pheromone in Eq. 1 which controls the movement of ants will be changed for each run. For GUIAnt-Miner, the pheromone information function has been defined as:

$$\tau_{ij}(t+1) = \frac{\tau_{ij}(t) + \tau_{ij}(t)Q}{\sum_{i=1}^m \sum_{j=1}^{n_i} \tau_{ij}(t)}, \quad (5)$$

where  $Q$  measures the quality of prediction rules over the training data set. This measure is defined as the product of the sensitivity and specificity:

$$Q = \frac{TP'}{TP' + FN'} \cdot \frac{TN'}{FP' + TN'}. \quad (6)$$

Note that  $TP'$  is the number of hosts covered by rule that has the class predicted by that rule,  $FP'$  is the number of hosts covered by rule that has a class different from the class predicted by that rule,  $FN'$  is the number of hosts that is not covered by rule but has the class predicted by that rule, and  $TN'$  is the number of hosts that is not covered by rule and that does not have the class predicted by that rule.

For the first iteration, the initial pheromone value is normally set to:

$$\tau_{ij}(t=0) = \frac{1}{\sum_{i=1}^m n_i} \quad (7)$$

After each iteration run, a result of the model can be expressed by a path of ant walking from a value of a feature through one of the other feature. However, this result does not specify to any target class yet and then cannot be utilized. We therefore check all hosts covered by the result from the training data set again to obtain a target class by majority vote, and subsequently create a rule as follows.

$$\text{IF } (A_i = a_{ix} \text{ AND } j = a_{jy} \text{ AND } \dots) \text{ THEN } (W = w_z)$$

Consequently, the iterative computation in Eq. 1 will terminate if it produces the set of rules covering all hosts in the training data set.

## 4 Experimental Results

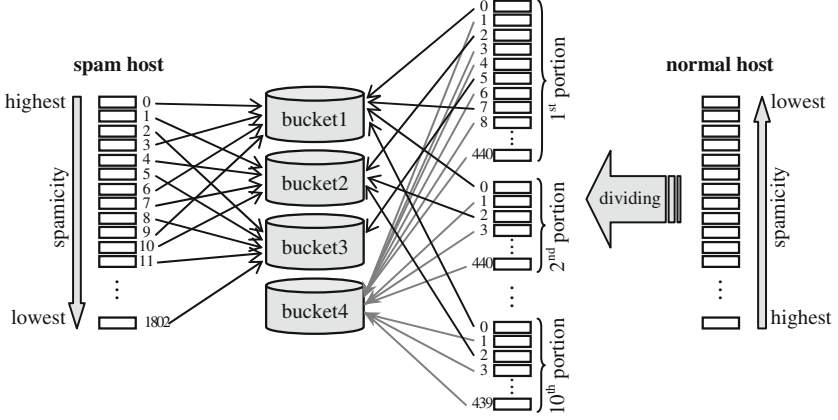
### 4.1 Data Set Preparation

We use the WEBSpam-UK2006 [9] containing hosts within .uk domain. From these, there are 1,803 hosts labeled as spam and 4,409 hosts labeled as normal. The data set contains several features including both content- and link-based features, as well as a spamicity value of each host. We further process this data set as follows (see Fig. 2):

- For the 1,803 spam hosts, we first sort them by ascending order of the spamicity values. Each host will be assigned with an identification number beginning from 0. We then decompose spam hosts into to 3 buckets by considering the remainder from dividing its identification number with 3. Eventually, we will have “bucket1”, “bucket2”, and “bucket3”, in which each contains equally 601 spam hosts.
- Similarly, for the 4,409 normal hosts, we sort them by descending order of the spamicity values. We equally divide them into 10 portions, and assign an identification number beginning from 0 to each host in each portion separately. For each portion, each identification number is again modulo by 7. The normal hosts whose remainder is 0, 2 and 5, will then be assigned into “bucket1”, “bucket2”, and “bucket3”, respectively. Note that the host with less identification number will be first assigned. To avoid data imbalance of normal and spam hosts in training set, we will stop the assigning process if each bucket contains 601 normal hosts. For all remaining hosts, we will put them into a new “bucket4”.

### 4.2 Host's Feature Selection

We use the information gain as a criterion to select the host's features. Figure 3 shows the 10 highest information gain features used to train the machine learning models. Of these, the first nine features are the link-based features; but only the last one is the content-based feature. Since all these features have continuous-range values, which cannot directly exploit in the GUIAnt-Miner program; we therefore discretize those values into 10 equal ranges.



**Fig. 2** Data preparation

- 
1. Logarithm value of TrustRank/PageRank of homepage
  2. Logarithm value of TrustRank/in-degree of homepage
  3. Logarithm value of TrustRank/PageRank of max PageRank
  4. Logarithm value of TrustRank of homepage
  5. Logarithm value of TrustRank/in-degree of max PageRank
  6. Logarithm value of TrustRank of max PageRank
  7. Logarithm value of number of different supporters (sites) at distance 4 from homepage
  8. Logarithm value of number of different supporters (sites) at distance 4 from max PageRank
  9. Logarithm value of number of different supporters (sites) at distance 3 from homepage
  10. Top 200 corpus recall (standard deviation for a ll pages in the host)
- 

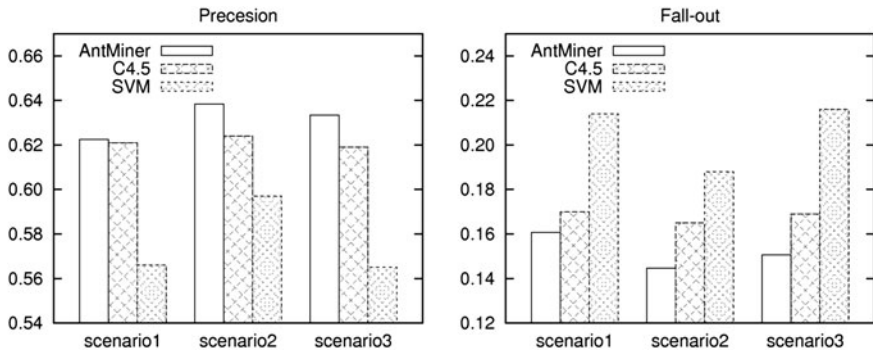
**Fig. 3** Features used to train the machine learning models

### 4.3 Results

From the set of data described in Sect. 4.1, we design 3 set of experiments according to the following scenarios:

- Scenario 1: we use bucket1 for training, while use bucket2, bucket3, and bucket4 for testing.
- Scenario 2: we use bucket2 for training, while use bucket1, bucket3, and bucket4 for testing.
- Scenario 3: we use bucket3 for training, while use bucket1, bucket2, and bucket4 for testing.

We compare performance of the ACO model with two other baselines, i.e., the decision tree (C4.5) and the support vector machine (SVM), using two standard measures: the positive predictive value (i.e., precision) and false positive rate (i.e., fall-out). To train the ACO model, we use 5 artificial ants. The terminating condition is either uncovered hosts by the rules are less than 10, or the number of



**Fig. 4** Spam host detection performance

iterations reaches 100. Generated rules that can cover at least 5 hosts will be kept as a candidate set of usable rules. These rules will finally be checked with the training data set again to obtain a target class by majority vote. For the C4.5 model, the rule pruning is disabled. For all other remaining parameters of C4.5 and SVM, the default setting in WEKA software [18] has been assigned.

The precision results in Fig. 4 show that the ACO learning model has the ability to detect spam hosts more accurate than C4.5 and SVM in all experiments. This is consistent with the fall-out results that the ACO learning model yields the least error prediction.

## 5 Conclusions

In this article, we propose to apply the ant colony optimization based algorithm to build a set of classification rules for spam host detection. Both content and link features extracted from normal and spam hosts have been exploited. From the experiments with the WEBSpam-UK2006 dataset, the proposed method provides higher precision in detecting spam than the basic decision tree C4.5 and SVM models. However, we currently just run our experiments using the default heuristic and basic pheromone updating function setting in the GUIAnt-Miner. In future work, we are looking forward to doing further experiments using other types of heuristic and pheromone updating functions, and hope to obtain higher quality set of classification rules.

## References

1. Gyöngyi Z, Garcia-Molina H (2005) Web spam taxonomy. In: Proceedings of the 1st international workshop on adversarial information retrieval on the web
2. Gyöngyi Z, Garcia-Molina H, Pedersen J (2004) Combating web spam with TrustRank. In: Proceedings of the 30th international conference on very large data bases



3. Wu B, Davison BD (2005) Identifying link farm spam pages. In: Proceedings of the 14th international world wide web conference
4. Dai N, Davison BD, Qi X (2009) Looking into the past to better classify web spam. In: Proceedings of the 5th international workshop on adversarial information retrieval on the web
5. Chung Y, Toyoda M, Kitsuregawa M (2009) A study of link farm distribution and evolution using a time series of web snapshots. In: Proceedings of the 5th international workshop on adversarial information retrieval on the web
6. Martinez-Romo J, Araujo L (2009) Web spam identification through language model analysis. In: Proceedings of the 5th international workshop on adversarial information retrieval on the web
7. Dorigo M, Di Caro G, Gambardella LM (1999) Ant algorithms for discrete optimization. *Artif Life* 5(2):137–172
8. Dorigo M, Maniezzo V, Coloni A (1996) Ant system: optimization by a colony of cooperating agents. *IEEE Trans Syst Man Cybern* 26(1):29–41
9. Castillo C, Donato D, Becchetti L, Boldi P, Leonardi S, Santini M, Vigna S (2006) A reference collection for web spam. *ACM SIGIR Forum* 40(2):11–24
10. Becchetti L, Castillo C, Donato D, Leonardi S, Baeza-Yates R (2006) Link-based characterization and detection of web spam. In: Proceedings of the 2nd international workshop on adversarial information retrieval on the web
11. Castillo C, Donato D, Gionis A, Murdock V, Silvestri F (2007) Know your neighbors: web spam detection using the web topology. In: Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval
12. Ntoulas A, Najork M, Manasse M, Fetterly D (2006) Detecting spam web pages through content analysis. In: Proceedings of the 15th international world wide web conference
13. Davison BD (2000) Recognizing nepotistic links on the web. In: Proceedings of AAAI workshop on artificial intelligence for web search
14. Henzinger MR, Motwani R, Silverstein C (2002) Challenges in web search engines. *ACM SIGIR Forum* 36(2):11–22
15. Fetterly D, Manasse M, Najork M (2004) Spam, dam spam, and statistics: using statistical analysis to locate spam web pages. In: Proceedings of the 7th international workshop on the web and databases
16. Page L, Brin S, Motwani R, Winograd T (1999) The PageRank citation ranking: bringing order to the web. Technical report, Stanford InfoLab
17. Internet archive. The wayback machine. <http://www.archive.org/>
18. Witten IH, Frank E (2005) Data mining: practical machine learning tools and techniques with Java implementations, 2nd edn. Morgan Kaufmann, San Francisco
19. Parpinelli RS, Lopes HS, Freitas AA (2002) Data mining with an ant colony optimization algorithm. *IEEE Trans Evol Comput* 6(4):321–332
20. Dorigo M, Gambardella LM (1997) Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Trans Evol Comput* 1(1):53–66
21. Dorigo, M (2004) Ant colony optimization public software. <http://iridia.ulb.ac.be/~mdorigo/ACO/aco-code/public-software.html/>

IT Convergence and Services

ITCS & IRoA 2011

Park, J.J.; Arabnia, H.R.; Chang, H.-B.; Shon, T. (Eds.)

2011, XXXVI, 700 p., Hardcover

ISBN: 978-94-007-2597-3