International Journal of Engineering Research and Development
eISSN : 2278-067X, pISSN : 2278-800X, www.ijerd.com
Volume 4, Issue 8 (November 2012), PP. 26-32

Comparison of Ant Colony and Bee Colony Optimization for Spam Host Detection

R. Sagayam[1], Mrs. K. Akilandeswari[2]
[1]*Research scholar Department of computer science Govt. Arts College (Autonomous) Salem-636007*
[2]*Asst. professor Department of computer science Govt. Arts College (Autonomous) Salem- 636007*

Abstract:—Web spam is the deliberate manipulation of search engine indexes.web spam involves a number of methods, such as repeating unrelated phrases, to manipulate the relevance or prominence of resources indexed in a manner inconsistent with the purpose of the indexing system. Search engine includes determining whether the search term appears in the content or URL of a webpage. We presents a spam host detection approach. The content and link features are extracted from hosts to train a learning model based on ant colony optimization (ACO) and bee colony optimization (BCO) algorithm. The dataset has been collected from WEBSPAM-UK2007 and implemented by java Environment. The optimal solution is compared with the ant colony and bee colony optimization. Finally, it provides which optimization algorithm is better in detecting spam.
Keywords:—Ant colony optimization, Bee colony optimization, content and link features and spam host detection

I. INTRODUCTION

Web search engine is designed to search for information on the World Wide Web. The search results are generally presented in a line of results often referred to as search engine result pages. The information may be a specialist in web pages, images, information and other types of files. Some search engines also mine data available in database or open directories. Unlike web directories, which are maintained only by human editors, search engines also maintain real-time information by running an algorithm on a web crawler. There are many search engine optimization methods that improve the quality and appearance of the content of websites and serve content useful to many users. Search engines use a variety of algorithms to determine relevancy ranking. These techniques involve altering the logical view that a search engine has over the page's contents. They all aim at variants of the vector space model for information retrieval on text collections. Keyword stuffing involves the calculated placement of keywords within a page to raise the keyword count, variety, and density of the page. This is useful to make a page appear to be relevant for a web crawler in a way that makes it more likely to be found. Link spam is defined as links between pages that are present for reasons other than merit.[14] Link spam takes advantage of link-based ranking algorithms, which gives websites higher rankings the more other highly ranked websites link to it. These techniques also aim at influencing other link-based ranking techniques such as the HITS algorithm. A common form of link spam is the use of link-building software to automate the search engine optimization process. Java environment includes a large number of development tools and hundreds of classes and methods. The development tools are part of the system known as Java Development Kit and the classes and methods are part of the Java Standard Library, also known as the Application Programming Interface. In Application Programming Interface, the Abstract Window Tool Kit package contains classes that implements platform-independent graphical user interface. Swing implements a new set of GUI components with a pluggable look and feel. It is implemented completely in java. Pluggable look and feel architecture allows to design a single set of GUI components that can automatically have the look and feel of any operating system platform. In this paper, we proposed to apply the ant and bee colony optimization algorithm in detecting spam host problem. The content and link based features extracted from normal and spam hosts have been used to train the classification model. The dataset is collected from WEBSPAM-UK2007 and has been implemented by java environment. The optimal solution is compared with the ant colony and bee colony algorithm in order to achieve the accuracy. The graph representation provides the comparison of ant and bee colony processing time in detecting spam.

II. LITERATURE SURVEY

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. 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 WEBSPAM-UK2006 [9], the results show that rules generated from ant colony optimization learning model can classify spam hosts more precise than the baseline decision tree (C4.5 algorithm) and support vector machine (SVM) models, that have been explored by many researchers [10–12]. We exploit the same intuition, in a slightly different way. It follows from the intuition of [13] that

it is also very unlikely for spam pages to be pointed to by good pages. Thus we start with a seed set of spam pages and propagate Anti Trust in the reverse direction with the objective of detecting the spam pages which can then be filtered by a search engine. We find that on the task of finding spam pages with high precision, our approach outperforms Trust Rank. We also empirically found that the average page-rank of spam pages reported by Anti-Trust rank was typically much higher than those by Trust Rank. This is very advantageous because filtering of spam pages with high page-rank is a much bigger concern for search engines, as these pages are much more likely to be returned in response to user queries.

III. ANT COLONY OPTIMIZATION

3.1 Basic concepts

In the natural world, ants (initially) wander randomly, and upon finding food return to their colony while laying down pheromone trails. If other ants find such a path, they are likely not to keep travelling at random, but to instead follow the trail, returning and reinforcing it if they eventually find food (see Ant communication). Overtime, however, the pheromone trail starts to evaporate, thus reducing its attractive strength. The more time it takes for an ant to travel down the path and back again, the more time the pheromones have to evaporate. A short path, by comparison, gets marched over more frequently, and thus the pheromone density becomes higher on shorter paths than longer ones. Pheromone evaporation also has the advantage of avoiding the convergence to a locally optimal solution. If there were no evaporation at all, the paths chosen by the first ants would tend to be excessively attractive to the following ones. In that case, the exploration of the solution space would be constrained. Thus, when one ant finds a good (i.e., short) path from the colony to a food source, other ants are more likely to follow that path.

3.2 Algorithm

The Ant colony optimization algorithm was aiming to search for an optimal path in a graph, based on the behavior of ants seeking a path between their colony and a source of food.

```
   procedure Ant colony optimization
Set Initialize parameters, pheromone trails
while (termination condition not met)
do
        Construct Ant Solution
        Update Pheromone Trails
        Daemon Actions
 end
 end
```

Fig.3.Ant Colony Optimization

The algorithm's scheme is shown in fig.3; after initializing the parameters and the pheromone trails, construct Ant Solution manages a colony of ants that concurrently and asynchronously visit adjacent states of the problem by moving through neighbor nodes of the problem's construction. They move by applying a stochastic local decision policy that makes use of pheromone trails and heuristic information. In this way, ants incrementally build solution to the problem. Once an ant has built a solution, or while the solution is being built, the ant evaluates the (partial) solution that will be used by update pheromone trails procedure to decide how much pheromone to deposit. Update Pheromones is the process by which the pheromone trails are modified, The trails value can either increase, as ants deposit pheromone on the components or decrease, due to pheromone evaporation. The deposit of new pheromone increases the probability that those components/connections that were either used by many ants or that were used by at least one ant and which produced a very good solution will be used again by future ants. Daemon actions procedure is used to implement centralized actions which cannot be performed by single ants. Examples of daemon actions are the activation of a local optimization procedure, or the collection of global information that can be used to decide whether it is useful or not to deposit additional pheromone to bias the search process from a non local perspective.

IV. BEE COLONY OPTIMIZATION

4.1 Basic concepts

A colony of honey bees can extend itself over long distances (up to 14 km) and in multiple directions simultaneously to exploit a large number of food sources. A colony prospers by deploying its foragers to good fields. In principle, flower patches with plentiful amounts of nectar or pollen that can be collected with less effort should be visited by more bees, whereas patches with less nectar or pollen should receive fewer bees[16][17][18][19]. The foraging process begins in a colony by scout bees being sent to search for promising flower patches. Scout bees move randomly from one patch to another. During the harvesting season, a colony continues its exploration, keeping a percentage of the population as scout bees. When they return to the hive, those scout bees that found a patch which is rated above a certain quality threshold (measured as a combination of some constituents, such as sugar content) deposit their nectar or pollen and go to the "dance floor" to perform a dance known as the waggle dance [16]. This dance is essential for colony communication, and contains three pieces of information regarding a flower patch: the direction in which it will be found, its distance from the hive and its

quality rating (or fitness). This information helps the colony to send its bees to flower patches precisely, without using guides or maps. Each individual's knowledge of the outside environment is gleaned solely from the waggle dance. This dance enables the colony to evaluate the relative merit of different patches according to both the quality of the food they provide and the amount of energy needed to harvest it. After waggle dancing inside the hive, the dancer (i.e. the scout bee) goes back to the flower patch with follower bees that were waiting inside the hive. More follower bees are sent to more promising patches. This allows the colony to gather food quickly and efficiently. While harvesting from a patch, the bees monitor its food level. This is necessary to decide upon the next waggle dance when they return to the hive. If the patch is still good enough as a food source, then it will be advertised in the waggle dance and more bees will be recruited to that source.

4.2 Algorithm

The Bees Algorithm is an optimization algorithm inspired by the natural foraging behavior of honey bees to find the optimal solution [15]. The algorithm requires a number of parameters to be set, namely: number of scout bees (n), number of sites selected out of n visited sites (m), number of best sites out of m selected sites (e), number of bees recruited for best e sites, number of bees recruited for the other (m-e) selected sites, initial size of patches which includes site and its neighborhood and stopping criterion.

Step1. Initialize population with random solutions.
Step 2. Evaluate fitness of the population.
Step 3. While (stopping criterion not met) //Forming new population.
Step 4. Select sites for neighborhood search.
Step 5. Recruit bees for selected sites (more bees for best e sites) and evaluate fit nesses.
Step 6. Select the fittest bee from each patch.
Step 7. Assign remaining bees to search randomly and evaluate their fit nesses.
Step 8. End While.

Fig.4.*Bee Colony Optimization*

In first step, the bees algorithm starts with the scout bees (n) being placed randomly in the search space. In step 2, the fit nesses of the sites visited by the scout bees are evaluated. In step 4, bees that have the highest fit nesses are chosen as "selected bees" and sites visited by them are chosen for neighborhood search. Then, in steps 5 and 6, the algorithm conducts searches in the neighborhood of the selected sites, assigning more bees to search near to the best e sites. The bees can be chosen directly according to the fit nesses associated with the sites they are visiting. Alternatively, the fitness values are used to determine the probability of the bees being selected. Searches in the neighborhood of the best e sites which represent more promising solutions are made more detailed by recruiting more bees to follow them than the other selected bees. Together with scouting, this differential recruitment is a key operation of the Bees Algorithm. However, in step 6, for each patch only the bee with the highest fitness will be selected to form the next bee population. In nature, there is no such a restriction. This restriction is introduced here to reduce the number of points to be explored. In step 7, the remaining bees in the population are assigned randomly around the search space scouting for new potential solutions. These steps are repeated until a stopping criterion is met. At the end of each iteration, the colony will have two parts to its new population – those that were the fittest representatives from a patch and those that have been sent out randomly [15].

V. DATA PREPROCESSING

Data preprocessing techniques can improve the quality of the data, thereby helping to improve the accuracy and efficiency of the subsequent mining process. Data preprocessing is an important step in the knowledge discovery process, because quality decisions must be based on quality data. Before processing the log, the data may be changed in several ways. For security or privacy reasons, the page addresses may be change into unique page identifications. This also will save storage space. Also, the data may be cleansed by removing irrelevant information. Large number of data sets may make the data mining process slow. Hence, reducing the number of data sets to enhance the performance of the mining process is important. Data pre-processing includes cleaning, normalization, transformation, feature extraction and selection, etc.

VI. EXPERIMENTAL ANALYSIS

6.1 Dataset preparation

We use the WEBSPAM-UK 2007 containing hosts within .uk domain. The data set is collected from WEBSPAM-UK 2007. The dataset is shown in fig.6.1.1

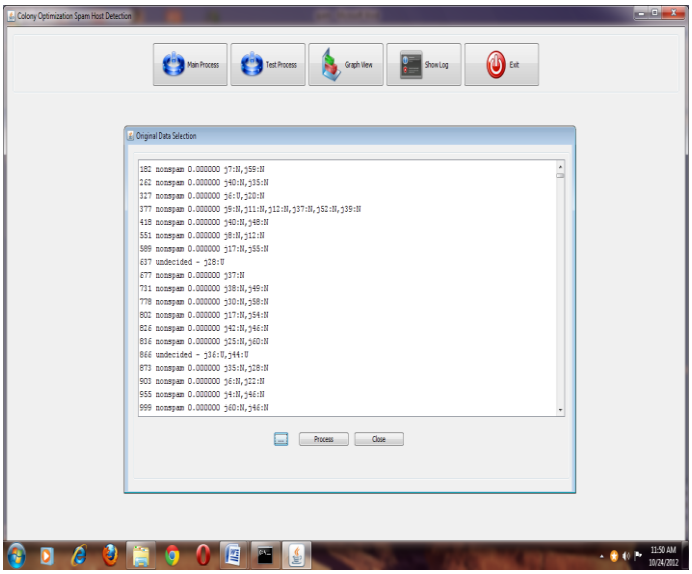

Fig.6.1.*1 Dataset*

It contains several features. The feature contains link and content that are extracted from host labeled as spam, non spam or normal. The dataset contains several data that are host id, host name, identification number and link id. The process of dataset is converted from a host id, host name and link id to URL id, URL type and link type. After that, the dataset will be preprocessed when link type is undecided. The dataset is created for training process. All the data is selected from dataset, to be detected by ant and bee colony optimization whether host feature is spam or normal.

6.2 Spam detection based ACO and BCO

The host features are selected from WEBSPAM-UK2007 dataset to be detected whether host features are spam or normal host based on Ant colony and bee colony optimization algorithm. It has implemented by java Environment. All the URL type and link type is selected and given to train to ant and bee to be detected whether host features are spam host or normal host in the data set. The training dataset is shown in fig.6.2.1 and fig.6.2.2

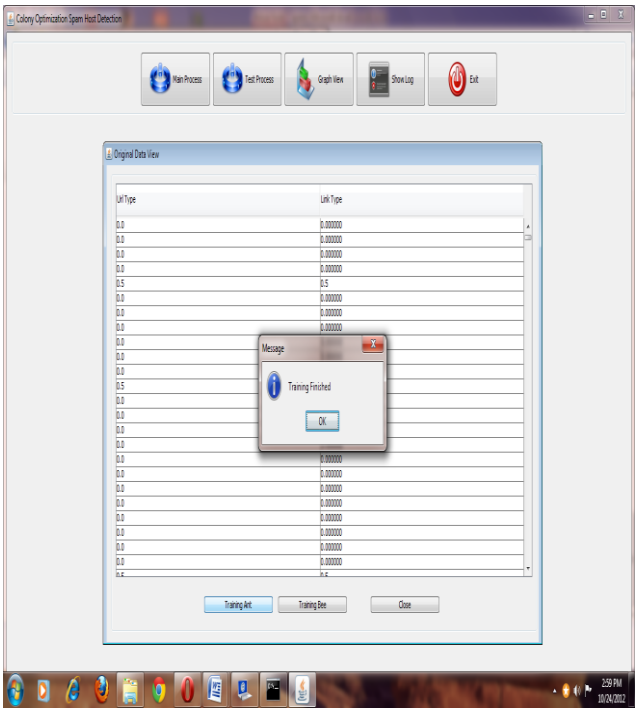

Fig.6.2.1. *Training dataset for ACO*

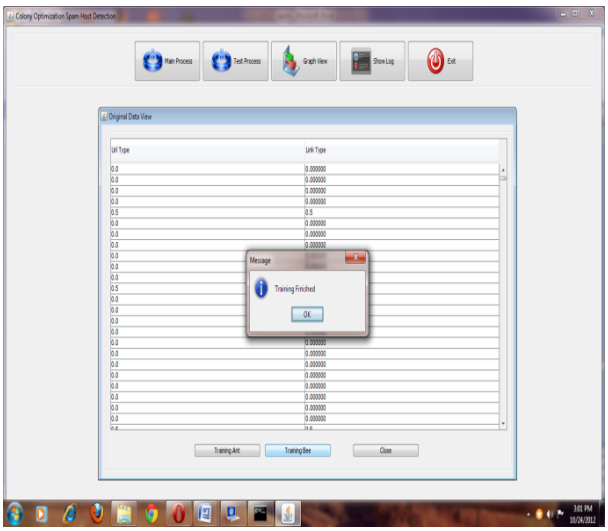

Fig.6.2.2. *Training dataset for BCO*

Ant and bee has followed three training rule or classification to be detected whether host features are spam or normal. If the process of data is selected and URL type and link type is 1 then the host feature is spam in the dataset. If the process of data is selected and URL Type and link type is 0 then the host feature is normal in the data set. If the process of data is selected and URL type and link type is greater than 0 and less than 0.5 then the host feature may be spam in the dataset. Every URL type and link type is trained whether host features are spam or normal by classification rule for each iteration in the dataset.

6.3 Results

All the URL type and link type is selected and tested for detecting whether host features are spam or normal by ant and bee colony optimization in the training dataset. Ant and bee is searched optimal distance (weights for how much URL type and link types are spam or normal) and calculated weights for each iteration. The testing dataset is shown in fig.6.3.1 and fig.6.3.2.

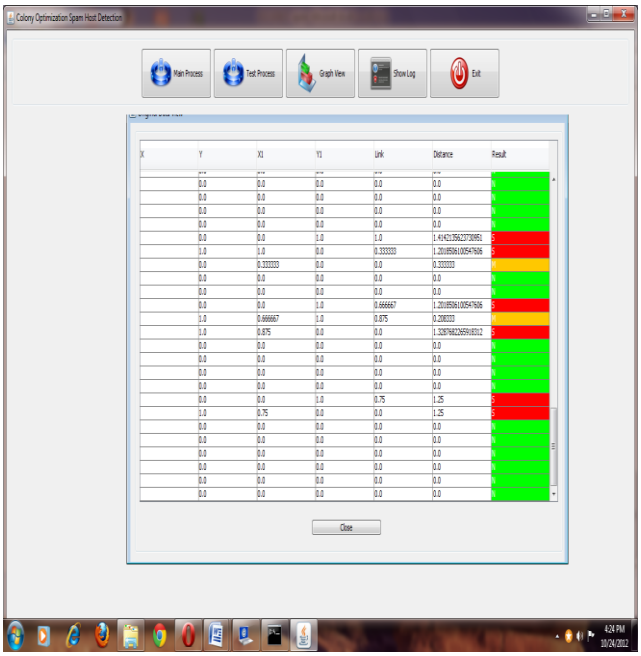

Fig.6.3.1 *Testing dataset for ACO*

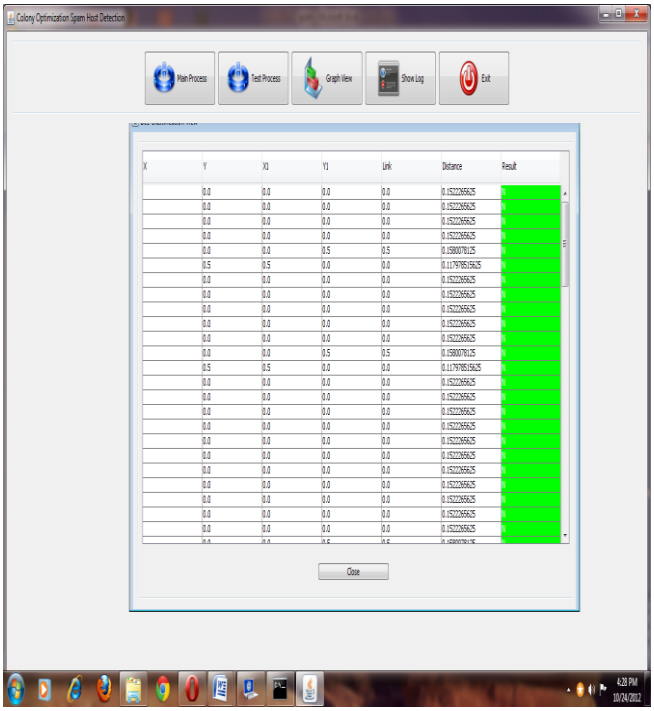

Fig.6.3.2 *Testing dataset for BCO*

Ant and bee has been followed three condition and labeled to be detected whether host features are spam and normal. If optimal distance is greater than or equal1 and greater than 0.5 then host features is spam and labeled as S and color representation of spam host is red. If optimal distance is zero then host features is normal and labeled as N and color representation of normal host is green. If optimal distance is greater than 0 and less than 0.5 then host features may be normal or spam and labeled as M and color representation of host feature is yellow. The processing time is compared with the ant colony and bee colony optimization. If the first process of ant colony and second process of bee colony is considered, then the processing time (Milliseconds) of bee colony optimization is less than the ant colony optimization. The first comparison graph is shown in fig.6.3.3

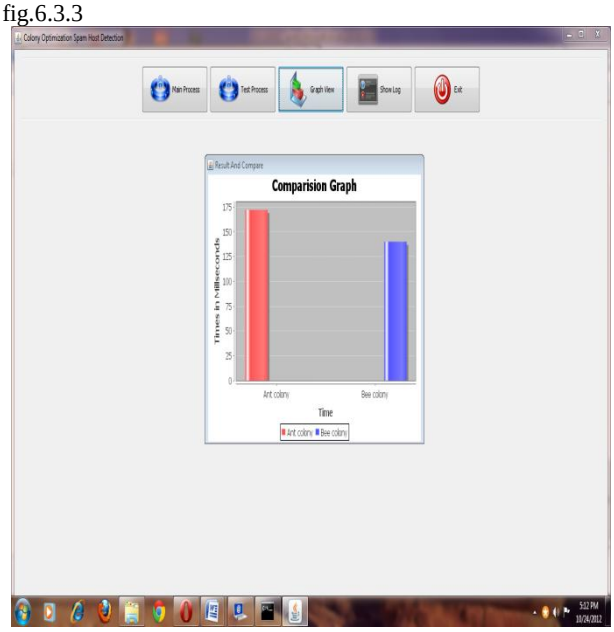

Fig.6.3.3 *The first comparison graph*

If the first process of bee colony and second process of ant colony is considered then the processing time (Milliseconds) of ant colony is less than the bee colony optimization. The second comparison graph is shown in fig.6.3.4.

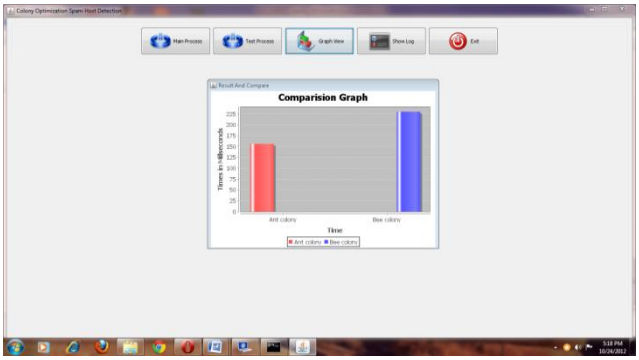

Fig.6.3.4 *The second comparison graph*

Ant colony Optimization obtained the solution (weight calculation for spam host detection) up to possible distance in the testing dataset, because ant provides optimal solution based on more number of pheromone trails. But bee colony Optimization obtained the solution (weight calculation for spam host detection) only particular distance, because bee provides optimal solution based on small number of trails.

VII. CONCLUSION

In this paper, we propose to apply the ant and bee colony optimization based algorithm to build a set of classification rule and comparison for spam host detection. The optimal solution (weight calculation for spam host detection) is compared with the ant colony and bee colony in the testing dataset. From the Experiments with the WEBSPAM-UK 2007 dataset, the ant colony optimization is higher performance based optimal solution than bee colony optimization. Therefore, the ant colony optimization is better based on algorithm in detecting spam. In future work, the same dataset is used and based on other types of heuristic information, and hope to determine better performance by the set of classification rule.

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, Kitsuregaw a 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]. Combating Web Spam with Trust Rank. Z. Gy¨ongyi, H. Garcia-Molina and J. Pedersen. In VLDB 2004.

[14]. Davison, Brian (2000), "Recognizing Nepotistic Links on the Web", AAAI-2000 workshop on Artificial Intelligence for Web Search, Boston: AAAI Press, pp. 23–28.

[15]. Pham D.T., Ghanbarzadeh A., Koç E., Otri S., Rahim S., and M.Zaidi "The Bees Algorithm – A Novel Tool for Complex Optimisation Problems"", Proceedings of IPROMS 2006 Conference, pp.454–461.

[16]. Von Frisch K. Bees: Their Vision, Chemical Senses and Language. (Revised edn) Cornell University Press, N.Y., Ithaca, 1976.

[17]. Seeley TD. The Wisdom of the Hive: The Social Physiology of Honey Bee Colonies. Massachusetts: Harvard University Press, Cambridge, 1996.

[18]. Bonabeau E, Dorigo M, and Theraulaz G. Swarm Intelligence: from Natural to Artificial Systems. Oxford University Press, New York, 1999.

[19]. Camazine S, Deneubourg J, Franks NR, Sneyd J, Theraula G and Bonabeau E. Self-Organization in Biological Systems. Princeton: Princeton University Press, 2003.