SEARCH AND OPTIMISATION WITH SMART ANT-LIKE SOFTWARE AGENTS

Weng Kin Lai, Kok Meng Hoe, Yit Mei Aw and Tracy Sock Yin Tai
Information Processing Research Group
MIMOS Berhad
Technology Park Malaysia
57000 Kuala Lumpur
Malaysia
email: lai@mimos.my
menghk@mimos.my
yitmei@mimos.my
sytai@mimos.my

ABSTRACT

Simple organisms that live in colonies, for example ants, bees, wasps and termites have long fascinated many people for their collective intelligence that is manifested in many of the things that they do.

Recently, computational paradigms based on the humble ant have been offered as a different approach to solve non-trivial engineering problems. This new computational paradigm replaces the traditional emphasis on control, preprogramming, and centralisation with designs featuring autonomy, emergence, and distributed functioning.

A growing community of researchers has applied such swarm intelligence to solve a variety of diverse applications. This paper describes the development of a pair of ant-based algorithms that were used to solve two very different engineering problems in search and optimisation.

Keywords: ***Swarm Intelligence, Artificial Intelligence, Classification, Web Document Clustering, Travelling Salesman Problem, and Combinatorial Optimisation***

1.0 INTRODUCTION

Besides ethologists, mathematicians and computer scientists have also proposed and developed models to explain the behaviour and capabilities of social insects that live in colonies. Ant algorithms and swarm intelligence systems have been offered as a novel computational approach that replaces the traditional emphasis on control, preprogramming, and centralisation with designs featuring autonomy, emergence, and distributed functioning.

The term "*swarm intelligence*" was first used by *Beni & Wang* in the context of cellular robotic systems, where many simple agents occupy one- or even two-dimensional environments to generate patterns and self-organise through nearest-neighbour interactions [1, 2, 3]. Today, the expression is used to include systems whereby the collective behaviours of (unsophisticated) agents interacting locally with their environment cause coherent functional global patterns to emerge. There are basically two very distinct and unique characteristics of these ants.

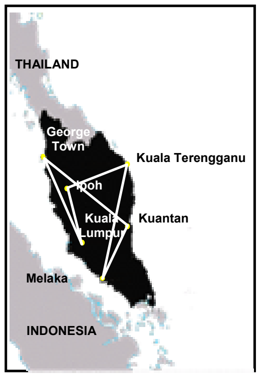
Controlled experiments to better understand the behaviour of the Argentine ant *Linepithema humile* have suggested that their ability to select the most efficient path from the nest to the food source is based on self-organisation within the colony [4]. On the other hand, several other species of ants are known to be very efficient and effective in organising their dead to form cemetries or to sort their larvae into several homogeneous piles [4]. Field observations on the workers of the *Leptothorax unifasciatus* ant have show that they systematically sort their colony's brood. Eggs and microlarvae are placed at the center of an area, while the pupae and prepupae are placed in the remaining areas from the center to the perimeter of their nest. It has been suggested that the ants pick up and drop items according to some knowledge of their surroundings. Similar *intelligent* behaviour in organising their dead have also been observed in the *Lasius niger* ants.

The rest of this paper is organised as follows. In Section 2, we will provide a brief introduction to the travelling salesman problem (TSP). Sections 3 to 6 describe some of the details on how the TSP may be solved with a swarm intelligence approach based on their foraging characteristics. Experimental results obtained with several sets of data will be shown.

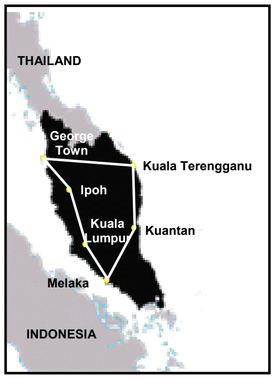
As we have suggested earlier, this foraging characteristic of the ants is but one of the useful behaviour that we can simulate, the other being the clustering dynamics seen in the *Messor sancta* and *Lasius niger* ants [5]. In Section 7, we will introduce the subject of web page classification. Some details of the smart ant-like algorithm that was used to automatically classify web documents are described in Section 8. Details of the experimental set-up to investigate the classification of English web pages with the smart ant-like agents are described in Sections 9, 10 and 11. The experimental results to automatically classify a set of web documents are shown in Sections 12 and 13. Finally, Section 14 concludes with a summary of the work described here.

2.0 THE TRAVELLING SALESMAN PROBLEM (TSP)

The *travelling salesman problem*, or TSP for short, is a well-documented and well-investigated problem [6]. Fig. 1 shows two possible tours for an example of a 6-city TSP. For such a "*small*" example the problem is easy to solve, but examples with a larger number of cities have shown that a systematic and exhaustive search for a solution is very computationally expensive.



(a) *An inefficient tour*



(b) *An efficient tour*

Fig. 1: An example of a 6-city TSP

3.0 ANT SYSTEM (AS) FOR TSP

For any TSP, the goal is to find a closed tour of minimal length connecting all the *n* given cities. Ant system (AS) generates solutions to the TSP by moving the ants on the graph from one city to another by exchanging information via the level of pheromone deposited on the edges of the graph until they make a complete tour [7]. During each iteration, each ant *k* ($1 \le k \le m$, where *m* defines the total number of ants used) would be executing a total of *n* steps to build a tour. Iteration *t* is in the range of $1 \le t \le t_{max}$ where t_{max} is the maximum number of iterations that is usually defined by the users.

At each iteration, the ants will move from city *i* to city *j*. They will make their decision based on,

- Whether the cities have been visited. A *memory* or *tabu list* is kept. It is associated with each ant so that no ant will visit a city more than once. It contains each ant *k* and a set J_i^k of the corresponding cities that it has not visited. It grows with the tour. When a tour is completed, the tabu list is emptied [7].
- Visibility, η_{ij}, is the inverse of distance, i.e. $\eta_{ij} = 1/d_{ij}$. It helps the ant to search the next shortest path, i.e. it will seek out those cities to visit that has a larger value for η_{ij}.
- The amount of virtual pheromone trail $\tau_{ij}(t)$ on the edge that connects city *i* to city *j*. It is updated as the tours are generated. This helps the ants to choose the next city *j* when the ant is at city *i*. Virtual pheromone trail is more global than the visibility η_{ij}, discussed previously.

The probability for an ant *k* to move from city *i* to city *j* while building the t^{th} tour is called the *random proportional transition rule* [4]. It is given as:

$$P_{ij}^{k}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{\lambda \in J_i^k}[\tau_{i\lambda}(t)]^{\alpha} \cdot [\eta_{i\lambda}]\beta}; & j \in J_i^k \\ 0; & j \notin J_i^k \end{cases} \tag{1}$$

α and β are two parameters that control the relative weight of the trail intensity $\tau_{ij}(t)$, and the visibility η_{ij} respectively. If $\alpha = 0$, the visibility plays a more important role, and the path with the shortest distance is more likely to be chosen. However, if $\beta = 0$, only the pheromone amplification is at work here. While this will lead to a rapid selection of the tour, unfortunately the tour itself may not be optimal. A tradeoff between tour length and the trail intensity is necessary if we want not only the short tours, but fast convergence of the algorithm as well.

After the completion of a tour, each ant will deposit a quantity of pheromone on each edge and the iteration counter *t*, will be increased by *1*. This is to indicate that the ant has already made a tour. The ant lays an amount of pheromone, $\Delta\tau_{ij}^{k}(t)$ on edge (*i,j*) according to the following equation,

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} Q/L^{k}(t) \text{ if } (i,j) \in \tau^{k}(t) \\ 0 \quad \text{if } (i,j) \notin \tau^{k}(t) \end{cases} \tag{2}$$

where $\tau^{k}(t)$ is the tour completed by ant *k* at iteration *t*. $L^{k}(t)$ is the tour length; while *Q* is a parameter that is usually set to be of the same magnitude as that of the expected optimal tour length.

Unfortunately, this does not work very well without *pheromone decay* [4]. In order to make it more efficient, the pheromone trail intensity should be allowed to decay. Otherwise, all the ants will end up with the same tour. This is commonly known as *stagnation.* Hence, trail decay is introduced by implementing a coefficient of decay ρ, where $0 \leq \rho < 1$ to the formulation. The resulting pheromone update rule, which is applied to all the edges, is:

$$\tau_{ij}(t) \leftarrow (1-\rho)\cdot \tau_{ij}(t) + \Delta\tau_{ij}(t) \tag{3}$$

where $\Delta\tau_{ij}(t) = \sum_{k=1}^{m}\Delta\tau_{ij}^{k}(t)$, and *m* is the number of ants. The initial amount of pheromone on the edges is assumed to be a small positive constant τ_0. The high-level description of AS may be found from [4].

4.0 IMPROVEMENTS TO ANT SYSTEM: THE ANT COLONY SYSTEM (ACS)

Improvements to the Ant System (AS) is based on the following modifications [4]:

(1) A different transition rule
(2) A different pheromone trail update rule
(3) The use of local updates of pheromone trail to favour exploration
(4) The use of candidate list to restrict the choice of next city to visit.

A more detailed description on the refinements is discussed below.

4.1 Transition Rule

The transition rule is now modified to encourage exploration. An ant *k* on city *i* chooses the city *j* to move based on the following rule:

$$J = \begin{cases} \arg\max_{u \in J_i^k} \left\{ [\tau_{iu}(t)] \cdot [\eta_{iu}]^{\beta} \right\}; & q \leq q_0 \\ J & ; q > q_0 \end{cases} \tag{4}$$

where q is a random variable uniformly distributed over [0,1]. q_0 is a tunable parameter ($0 \leq q_0 \leq 1$), and $J \in J_i^k$ is a city that is randomly selected according to the probability,

$$P_{ij}^{k}(t) = \frac{[\tau_{iJ}(t)] \cdot [\eta_{iJ}]^{\beta}}{\sum_{l \in J_i^k} [\tau_{il}(t)] \cdot [\eta_{il}]^{\beta}} \tag{5}$$

The ACS transition rule is identical to AS when $q > q_0$, a condition which favours more exploration. However, when $q \leq q_0$, the system now encourages the exploitation of the knowledge available about the problem. In other words, it is based on the heuristic knowledge about the distance between cities and the learned knowledge memorised from the pheromone trails.

This parameter allows the system to concentrate on the best solution instead of letting it to explore aimlessly.

4.2 Pheromone Trail

In the earlier implementation [8], all the ants were allowed to deposit pheromone after they have completed their tours. However, in ACS the global pheromone trail updating rule is applied only to the edges belonging to the best tour since the beginning of the trail. The updating rule is:

$$\tau_{ij}(t) \leftarrow (1-\rho) \cdot \tau_{ij}(t) + \rho \Delta \tau_{ij}(t) \tag{6}$$

$$\Delta \tau_{ij}(t) = 1/L^{+} \tag{7}$$

where (i, j) is the edge belonging to T^{+}, the best tour found since the beginning of the trail, while ρ is a decay parameter.

4.3 Local Updates of Pheromone Trail

When performing a tour, ant k is in city i and selects city $j \in J_i^k$. The pheromone concentration will be updated as follows:

$$\tau_{ij}(t) \leftarrow (1-\rho) \cdot \tau_{ij}(t) + \rho \cdot \tau_0 \tag{8}$$

The value τ_0 is the same as the initial value of pheromone trails and it was experimentally found that setting

$$\tau_0 = (n \cdot L_{nn})^{-1} \tag{9}$$

produces good results. L_{nn} is the length of a tour produced by the nearest neighbour heuristic.

The role of this local update of pheromone trail is to shuffle the tours so that the cities selected in one of the ant's earlier tour may be explored later by the other ants as part of another tour. This makes the learned desirability of the edges change dynamically. Without this local update, all the ants would be constrained to search in a narrow neighbourhood of the best previous tour.

4.4 Use of a Candidate List

A candidate list keeps track of the preferred cities to be visited and these cities are arranged in an ascending order of the distances from the present city. An ant will first restrict the choice of the next city to those in the candidate list and will only consider the other cities (not in the candidate list) if all the cities in the candidate list have already been visited.

5.0 EXPERIMENTAL SETUP

Such an ant-based computational algorithm has been simulated on several data sets of the TSP. These consist of TSPs with 22, 48 and 100 cities, as shown in Fig. 2, Fig. 3, and Fig. 4 respectively.

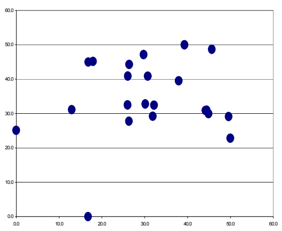


Fig. 2: 22-city TSP

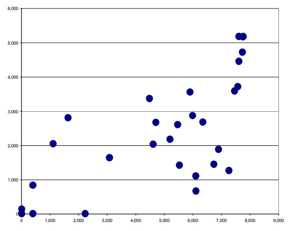


Fig. 3: 48-city TSP

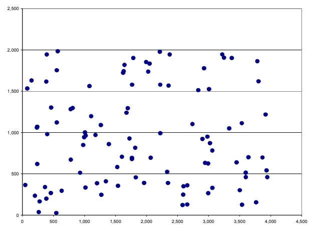


Fig. 4: 100-city TSP

5.1 Number of Ants

In our investigation, the total *number of ants* was set equal to the number of cities. Too many ants would quickly reinforce the sub-optimal trails, thereby leading to premature convergence to weak solutions. On the other hand, too few ants would not produce the expected synergistic effects of cooperation because of the process of pheromone decay.

5.2 Parameters Used

For AS, $\alpha = 1$, $\beta = 5$, $\rho = 0.5$, $Q = 100$, $\tau_0 = 10^{-6}$ and $e = 5$, while $\beta = 2$, $\rho = 0.1$, $q_0 = 0.9$, $m = 10$, $\tau_0 = (n \cdot L_{nn})^{-1}$ for ACS.

6.0 RESULTS AND DISCUSSION

When the size of the TSP scales up from a smaller number of cities [9] to as much as 100 cities, it may be evident that ACS always generates the shortest tours. Moreover, the non-optimum tours are also quite close to the optimum ones as well. This may be due to the improvements to the basic swarm intelligence approach in not only using the pheromone behaviour, but also leveraging on some inherent knowledge of the problem. Independent studies in using conventional techniques to solve the TSP have also shown that the solutions generated by the swarm-based algorithm are not inferior in any way to them for the various data sets investigated [4, 9].

Table 1: Experimental Results

Number of Cities	Method	Shortest Total Distance	Average	Standard Deviation
22	ACS	78.74	78.80	0.82
	AS	86.90	91.29	4.01
48	ACS	36,986.39	36,995.00	190.82
	AS	39,236.88	42,159.43	1,451.86
100	ACS	24,336.97	24,348.07	183.11
	AS	24,726.45	27,417.90	1,523.33

In the next few sections we will illustrate how another very distinctive behaviour of the ants may be used to solve a significantly different type of problem.

7.0 WEB DOCUMENT CLASSIFICATION

As the amount of online information in the form of web pages grows, the demand of text categorisation to assist in efficient retrieval will surely increase in tandem. Even though such a task may be performed manually by the domain experts, it is unlikely that such human categorisation will be able to keep pace with the tremendous growth of the World Wide Web (WWW). The amount of scientific information and the number of electronic journals on the Internet are also expected to expand from the 1,000 journals reported in 1996 [10]. A study by *Cyveillance* estimates that there is a total of 2.1 billion unique, publicly accessible web pages [11]. Consequently, the sheer volume of data that is available on the Internet will surely overwhelm these human categorisers. Hence, as the Internet expands, the importance of having this process automated will become increasingly important.

Clustering techniques have been commonly used to retrieve, filter, and categorise documents. The traditional clustering algorithms either use a priori knowledge of document structures to define a distance or similarity among these documents, or use probabilistic techniques such as Bayesian classification. A significantly different approach was adopted by *Tai, Suit & Eng* [12] in which the HTML tags associated with each web page were used to categorise them.

In recent studies, several species of ants have been observed to cluster their dead to form a cemetery. Based on the work of *Deneubourg et al*. [13] the sorting behaviour of such ants have been used for various applications, ranging from exploratory data analysis to graph partitioning [4]. This part of the paper investigates how this corpse clustering behaviour found in ants may be extended to classify English web documents.

8.0 WEB DOCUMENT REPRESENTATION

The clustering process of documents, in this case, web pages, involves implementing suitable clustering techniques to group together documents that have similar characteristics. However, before the similar documents can be grouped together, an important process is to identify and extract all the relevant features of each document. The feature extraction of a document basically involves finding the representation of the word vector or set of descriptors that best describe it. Concise representations are usually derived from the contents of a more complex objects. In the case of textual objects i.e. documents (more specifically web pages), words taken directly from the document are augmented with weights and traditionally are used to form a *bag-of-words* representation disregarding the linguistic context variation at the morphological, syntactical, and semantically levels of natural language.

8.1 Automated Text Processing

Automated text processing is the process of producing document representations or "*bags of words*" (also known as index terms) automatically. Conventionally, text processing follows a standard procedure, which may be mainly divided into 4 major text operations [14] which will be described further in the next few sections.

8.1.1 Lexical Analysis

This is generally defined as the process of converting a stream of characters (the text of the documents) into a stream of words (the candidate words to be adopted as index terms and it involves more than "*linear analysis*" or "*scanning*" of spaces between the words as word separators. The stream of characters making up the text is read one at a time and grouped into *lexemes (*lexemes are minimal lexical unit of a text). There are four particular cases that need to be considered with care, viz.

- Digits
- Hyphens
- Punctuation marks
- Letters

8.1.2 Stopwords Elimination

Stopwords are very commonly used words, and in the English language these would be articles, pronouns, adjectives, adverbs and prepositions, that are known to make poor index terms. They are usually removed from further consideration as index terms when identified in a document. The process of stopwords elimination is illustrated in Fig. 5 with part of Dr. *Martin Luther King Jr.'s* well-known "*I Have a Dream"* speech that he delivered on the steps of the Lincoln Memorial in Washington D.C. on August 28, 1963.

I have a dream that one day this nation will rise up and live out the true meaning of its creed: We hold these truths to be self-evident: that all men are created equal. I have a dream that one day on the red hills of Georgia the sons of former slaves and the sons of former slave owners will be able to sit down together at a table of brotherhood. I have a dream that one day even the state of Mississippi, a desert state, sweltering with the heat of injustice and oppression, will be transformed into an oasis of freedom and justice. I have a dream that my four children will one day live in a nation where they will not be judged by the color of their skin but by the content of their character. I have a dream today.

Total Word Count : *143*

(a) *The Original Text*

dream day nation rise live true meaning creed: hold truths self-evident: created equal. dream day red hills Georgia sons former slaves sons former slave owners able sit table brotherhood. dream day Mississippi, desert state, sweltering heat injustice oppression, transformed oasis freedom justice. dream children day live nation judged color skin content character. dream today

Total Word Count : *54*

(b) *With stopwords removed*

Fig. 5: Stopwords Elimination

Notice that there is now a significant reduction in the number of words amounting to about 62.2% of the total initial amount after the stopwords elimination process. If we were to have a closer look at just the first sentence of the address shown here, we can clearly see that common words like *I, have, a, that, one, this, will, up, and, out, the, of,* and *its*, have been eliminated as stopwords.

8.1.3 Stemming

The objective of stemming [15] is to remove affixes (i.e *prefixes* and *suffixes*) so as to reduce the total size of the index terms. This is normally done with ways of finding morphological variants of terms in documents. Examples of the stemmed index terms are shown in Table 2 below.

Table 2: Example of the stemming process on several terms

Original Term	**After stemming**
possibilities	*possibl*
Possible	*possibl*
possibility	*possibl*
Possibly	*possibli*
Software	*Softwar*
Software	*softwar*

8.1.4 Indexing

This is the final process of text processing where the index terms are extracted to identify the features for each document. Feature extraction of a document involves finding the optimal representation of the word vector or set of descriptors that best describe the salient features of the documents.

9.0 ARTIFICIAL ANT COLONIES FOR EXPLORATORY DATA ANALYSIS

Data clustering, or just *clustering*, is an explorative task that seeks to identify groups of similar objects based on the values of their attributes [16]. Clustering works on the inherent characteristics of the data and attempts to discover distinct groupings or boundaries to divide the data set into meaningful partitions. *Deneubourg et al*. [13] proposed a basic model which generalised the clustering behaviour of ants into two simple actions, viz.

i) picking up an isolated item, and
ii) dropping the item when a larger number of similar items are present in the neighbourhood.

Assuming the ants can pick up only one item at a time and only one type of item exists in the environment, each action may hence be defined in terms of a pair of probabilistic functions as shown below:

$$\text{Picking up probability, } P_p^a = \left(\frac{k_1}{k_1 + f} \right)^2 \tag{10}$$

$$\text{Dropping probability, } P_d^a = \left(\frac{f}{k_2 + f} \right)^2 \tag{11}$$

f is the density of item a in the neighbourhood of the agent, while k_1 and k_2 are threshold constants. The picking up probability, P_p^a is high when $f \rightarrow 0$, i.e. the density of item a in the vicinity of the ant is low. Inversely, P_d^a is low when $f \rightarrow 0$, i.e. the density of item a surrounding the individual ant is low. As such, the item will be moved until the ant reaches a denser region.

Deneubourg's [13] model was extended by *Lumer & Faieta* [17] to include a distance function, d between the data objects. This removed the need for assuming type homogeniety and making it more generalised for the purposes of exploratory data analysis. They provided an algorithm, which models the inherent similarity of data objects onto a lower dimensional space, e.g. 2-D. In their algorithm, the imitation ants move randomly on a grid and decide to pick up (if unladen) or drop (if laden) an object based on a local density function, $f(o_i)$. This function measures the average similarity of the discovered object i with other objects j in its perceivable neighbourhood (e.g. 8 adjacent cells for a square $[n \times n]$ grid). Given a constant α, $d(o_i, o_j)$ as the distance between object i and j, and *Neighbourhood*(c) as the number of cells adjacent to the one occupied by the ant at any time t, $f(o_i)$ is defined as follows:

$$f(o_i) = \frac{1}{m^2} \sum_{o_j \in neighbourh\,ood} \left[1 - \frac{d(o_i, o_j)}{\alpha} \right] \text{ if } f > 0, \qquad f(o_i) = 0 \text{ otherwise.} \tag{12}$$

Other similar work includes the *AntClass* [18] clustering algorithm, which is a combination of an ant colony with the partitional *K-Means* algorithm. The ant colony of *AntClass* differs from *Lumer & Faieta's* [17] model as the ants are allowed to carry more than a single object at a time, have local memory and other heterogeneous features. *Monmarché et al.* [18] applied their hybrid algorithm to six different real-world data sets and achieved an average misclassification rate of 5.9% within efficient running times.

Here we employed a colony of homogenous ant-like agents similar to *Lumer & Faieta* [17], but with a reformulation of the local density function to use a *similarity* instead of a distance measure. We then applied our algorithm (defined in the next section) to the non-trivial problem of clustering a collection of multi-class English web pages obtained from popular Internet search engines.

10.0 HOMOGENEOUS MULTI-AGENT SYSTEM FOR WEB PAGE CLUSTERING

Our multi-agent system consists of three main components: i) the colony of ant-like agents, ii) the items for clustering (in this case the feature vector of web pages), and iii) a square 2-D discrete space or grid, which we call the *clustering workspace*. Note that we will be using the term *agent* and *ant* interchangeably for the rest of this paper. The ant-like agents do not fall off the *toroidal* clustering space and moves only one step in any direction at any time unit, i.e. moving from its original cell to an *unoccupied* adjacent cell. Only a single agent and/or a single object is allowed to occupy a cell at a time. A single ant occupying cell c on the clustering space immediately perceives a neighbourhood of 8 adjacent cells, i.e. the *Neighbourhood*(c) for this ant is 8. However, when an *unladen* agent encounters an object o_i at cell c, it will have to decide to either pick up or ignore this object o_i with a probability P_p based on a local density function, $g(o_i)$, which determines the similarity between o_i and other objects o_j, where $j \in$ *Neighbourhood*(c). On the other hand if an ant that is already *laden* with object o_i lands on an empty cell c, it then calculates a probability P_d based on the same function $g(o_i)$ and decides whether to drop o_i or keep on carrying it. The function $g(o_i)$ is defined as follows:

$$g(o_i) = \frac{1}{neighbourhood(c)} \sum_{o_j} s(o_i, o_j) \tag{13}$$

where s is the similarity between two objects o_i and o_j.

For web page clustering we used the *cosine* similarity measure which may be defined as,

$$s_{\cos ine}(doc_i, doc_j) = \sum_{k=1}^{r} f_{i,k} \times f_{j,k} \Big/ \sqrt{\sum_{k=1}^{n} (f_{i,k})^2} \times \sqrt{\sum_{k=1}^{m} (f_{j,k})^2} \tag{14}$$

where m is the total number of matched terms between web pages doc_i and doc_j, and f is the frequency of matched terms between doc_i and doc_j. r is the total number of common terms in both documents.

The picking up P_p probability and dropping, P_d probability of the ants is as shown below.

$$P_p(o_i) = \left(\frac{k_1}{k_1 + g(o_i)} \right)^2 \tag{15}$$

$$P_d(o_i) = \begin{cases} 2g(o_i) \ , & if\ g(o_i) < k_2 \\ 1 \ , & if\ g(o_i) \geq k_2 \end{cases} \tag{16}$$

A description of our multi-agent system in algorithmic form is given in Fig. 6.

```
/* Initialisation */
For every item $o_i$
    Put $o_i$ on a randomly chosen empty cell of grid
Endfor
For every agent $a_i$
    Put $a_i$ on a randomly chosen unoccupied cell of grid
Endfor
/* End of Initialisation */

/* Main Loop */
For $t$ i n 1 to $t_{MAX}$ do
    For every agent $a_i$ do
        Move $a_i$ to a randomly selected cell, $c$
        If ( $c$ is non-empty with $o_i$ ) and ( $a_i$ is unladen )
            Find neighborhood of $c$
            For all items $o_j$ in the neighborhood of $c$
                Compute and sum $s(o_i,o_j)$
            Endfor
            Compute $f(o_i)$ using the sum of $s(o_i,o_j)$
            Compute $P_p(o_i)$ using $f(o_i)$
            Randomly pick real number $r$ in between 0 and 1
            If ($P_p(o_i) > r$)
                $a_i$ picks up object $o_I$
            Endif
        Endif
        If ($c$ is empty) and ($a_i$ is laden with $o_i$)
            Find neighborhood of $c$
            For all items $o_j$ in the neighborhood of $c$
                Compute and sum $s(o_i,o_j)$
            Endfor
            Compute $f(o_i)$ using the sum of $s(o_i,o_j)$
            Compute $P_d(o_i)$ using $f(o_i)$
            Randomly pick real number $r$ in between 0 and 1
            If ($P_d(o_i) > r$)
                $a_i$ drops object $o_i$
            Endif
        Endif
    Endfor
Endfor
/* End of Main Loop */
```

Fig. 6: Algorithm of the multi-ant-like agent system for clustering

11.0 INPUT DATA REPRESENTATION AND PRE-PROCESSING

For the experimental data, we identified 84 web pages from 4 different categories—*Business, Computer, Health* and *Science*, which resulted in a collection of 17,776 distinct words. To reduce memory requirements during clustering, the collection was represented by a sparse matrix with three elements per row: (i) a unique web page identifier, (ii) a unique word identifier, and (iii) the frequency value of each word within the web page. This is illustrated in Fig. 7.

```
<page ID>,  <wordID>,  <wordFreq>
<page ID>,  <wordID>,  <wordFreq>
<page ID>,  <wordID>,  <wordFreq>
<page ID>,  <wordID>,  <wordFreq>
…
```

Fig. 7: Sparse matrix representation of web pages

One of the inherent challenges of representing these documents with a set of keywords involves the optimal selection of these words for the set of features. Using all the *17,776* words will no doubt leave all the documents represented, but the sheer size of the set of words will incur a very heavy computation overhead. Even then, many of these words do not occur across many of the documents. It is rare to have all the words in the list to occur throughout the whole set of documents. This is illustrated in Fig. 8 with a sub-set of the 8 documents for each category with 20 words. The Y-axis represents the frequencies of the words for each web document. Even though this figure shows only 2 documents from each category it clearly illustrates the occurrences of the words and their frequencies for each document.

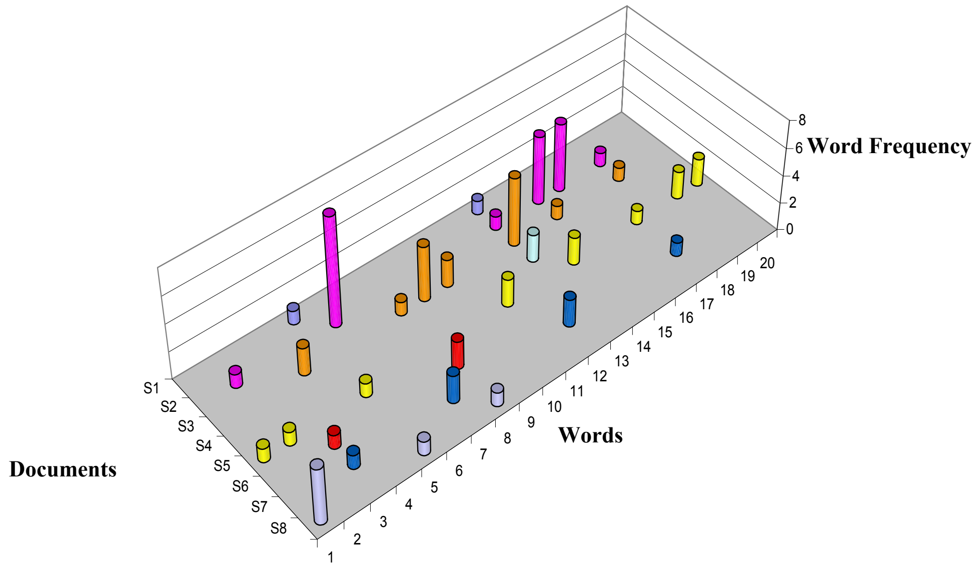


Fig. 8: 3D representation of the feature coverage for 8 documents

Intuitively, it would be good to remove all those features (words) that have very low occurrences across the documents in the set. Good candidates for such a removal would be those that occur in all the documents in any one class of similar documents.

12.0 EXPERIMENTAL SETUP

We used a 30×30 toroidal grid for the clustering workspace and 15 homogeneous ant-like agents.

Hence, all the cells that are lying at the perimeter of this workspace will be adjacent to each other. For example, the cells in column ***1*** of a 30 × 30 workspace will have neighbours in columns ***2*** and ***30***, as shown in Fig. 9 above. Similarly, the cells in the top **row a**, will have **row b** and ***row ad*** as their neighbours.

The maximum number of iterations, t_{MAX} for the algorithm was set to 500,000. We ran the algorithm numerous times using different values of k_1 (Eq. 15) and k_2 (Eq. 16) that ranges from 0.01 to 0.2, with incremental steps of 0.05.

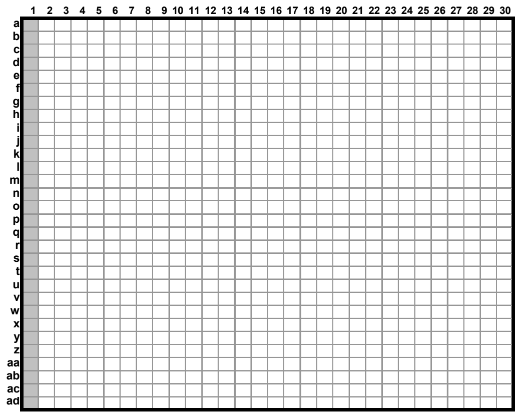
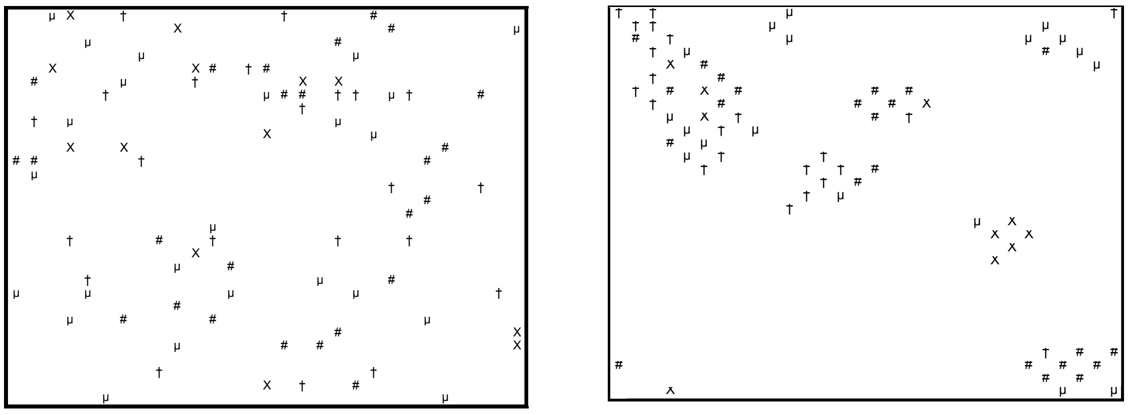


Fig. 9: A 30×30 toroidal workspace

13.0 RESULTS AND DISCUSSION

As shown in Fig. 10(a) for *t* = 0, the 84 web documents were randomly distributed on various cell locations within the workspace. A total of 15 ant-like agents have been used here, but they are not shown on these maps. After 50,000 iterations (see Fig. 10(b)), clusters of mixed-categories of web documents were formed. Since the workspace is continuous, clusters occupying the upper-left and bottom-right regions can be considered as a single cluster. The size of clusters varied from a minimum size of 6 documents to a maximum of about 40.

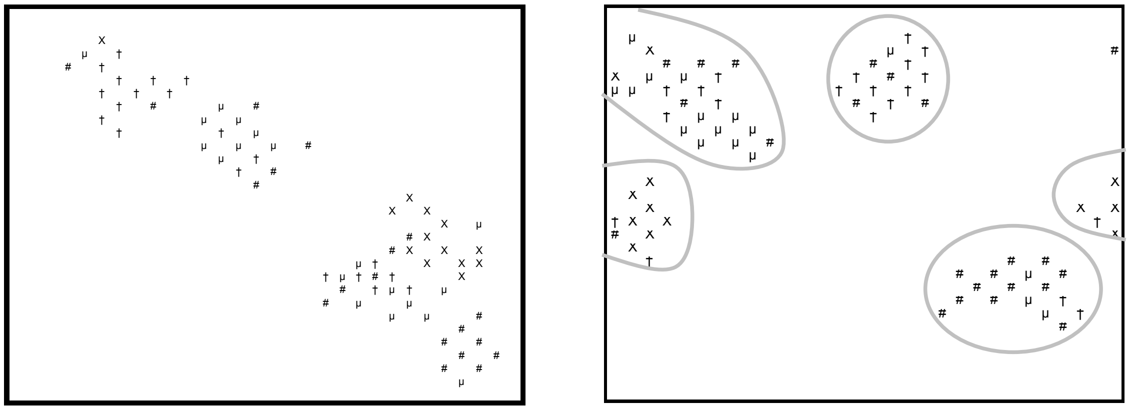


(a) *t* = 0 (b) *t* = 50,000

Fig. 10(a-b): Two-dimensional spatial distribution of web pages on a 30×30 toroidal grid at different time steps, t. The different categories of web pages are marked with symbols † for Business, X for Computers, μ for Health and # for Science

The best results were found at *t* = 300,000, where four distinct clusters of near-homogeneous type were formed on the workspace (see Fig. 10(d)). In other words, the majority of documents in each cluster originated from a single category, and web documents of different categories were grouped into different clusters. In addition, the number of similar-sized clusters equaled the actual number of available web document categories.

The automatic organisation of web pages in large databases into meaningful groups or classes would be an immensely useful tool for the efficient retrieval and management of such information. In the second part of this paper, we presented the findings of our study on using a multi-agent system based on the behaviour of social insects i.e. ants, in classifying web pages retrieved from popular web search engines. Unlike earlier works based on statistical techniques, we introduced the direct application of a similarity measure for determining the local density of web pages, which is more efficient than computing the density based on distance measures. Although the results obtained do not show web page clusters that are equaled to the classification ability of human experts, we believe there are still many research opportunities to improve the current system.



(c) t = 200,000 (d) t = 300,000

Fig. 10(c-d): Two-dimensional spatial distribution of web pages on a 30×30 toroidal grid at different time steps, t. The different categories of web pages are marked with symbols † for Business, X for Computers, μ for Health and # for Science

14.0 CONCLUSIONS

The purpose of this paper is to present a class of optimisation algorithm inspired by some of the simple organisms that live in colonies. Each of these insects in the colony seems to have its own private agenda, and yet the group as a whole appears to be highly organised. Moreover, the seamless integration of all individual activities does not seem to require any supervision and yet as a group they function exceedingly well for their individual simplicity.

In this paper we have introduced two very different aspects of the behaviour of some species of ants. These have been used to find the shortest path for a travelling salesman as well as to cluster a set of web documents. Experimental results obtained from each approach confirmed the potential of swarm intelligence to solve such problems. In the future we will explore how swarm intelligence may be used in other areas and problem domains.

REFERENCES

[1] G. Beni, "The Concept of Cellular Robotic System", in *Proceedings 1988 IEEE International Symposium on Intelligent Control, Los Alamitos*, *CA: IEEE Computer Society Press*, 1988, pp. 57-62.

[2] G. Beni and J. Wang, "Swarm Intelligence", in *Proceedings Seventh Annual Meeting of the Robotics Society of Japan, Tokyo: RSJ Press*, 1989, pp. 425-428.

[3] G. Beni and J. Wang, "Theoretical Problems for the Realization of Distributed Robotic Systems", in *Proceedings 1991 IEEE International Conference on Robots and Automation, Los Alamitos, CA: IEEE Computer Society Press*, 1991, pp. 1914-1920.

[4] E. Bonabeau, M. Dorigo and G. Theraulaz*, Swarm Intelligence From Natural to Artificial Systems*, Oxford University Press, 1999.

[5] R. Beckers, J. L. Deneubourg and S. Goss, "Trails and U-Turns in the Selection of the Shortest Path by the Ant Lasius Niger", in *Journal of Theoretical Biology*, Vol. 159, 1992, pp. 397-415.

[6] E. L. Lawler and A. H. Rinnooy-Kan, *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*. John Wiley & Sons; September 1985.

[7] M. Dorigo and L. M. Gambardella, "Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem". *Tech. Report TR/IRIDIA/1996-5*, Universite Libré de Bruxelles, Belgium.

[8] Lai W. K., Lim Keh Long and Aw Yit Mei, “Understanding Swarm Intelligence in Solving Combinatorial Optimization Problems”, *Proceedings of the Third MIMOS R&D Symposium on ICT and Microelectronics*, 8th November 2001.

[9] Y. M. Aw and W. K. Lai, “Investigations into the Use of Swarm Intelligence to Solve Combinatorial Optimization Problems”, in *Proceedings of the International Conference on Artificial Intelligence in Engineering and Technology*, 17th and 18th June 2002, Kota Kinabalu, Sabah, Malaysia, pp. 481-486.

[10] S. Lawrence and C. L. Giles, “Searching the World Wide Web”, in *Science*, Vol. 280 (5360) (1998), pp. 98-100.

[11] Cyveillance, “Sizing the Internet: A Cyveillance Study”, 10th July 2000.

[12] T. Tai, W. Y. Suit and N. L. Eng, “An Investigation of Web Page Classification Based on HTML Tags”, in *Proceedings Third MIMOS R&D Symposium on ICT & Microelectronics, Kuala Lumpur, Malaysia*, 2001.

[13] J. L. Deneubourg et al*., The Dynamics of Collective Sorting: Robot-Like Ants and Ant-Like Robots*, in Simulation of Adaptive Behaviour: From Animals to Animats, Meyer, J-A & Wilson, S. (eds), Massachusetts, MIT Press, 1990, pp. 356-365.

[14] R. Baeza-Yates and B. Ribeiro-Yates, *Modern Information Retrieval*. New York, ACM Press, 1999.

[15] M. F. Porter, “An Algorithm for Suffix Stripping”, in *Program*, Vol. 14 (3) (1980), pp. 130-137.

[16] J. A. Hartigan, *Clustering Algorithms*, New York, Wiley, 1975.

[17] E. D. Lumer and B. Faieta, “Diversity and Adaptation in Populations of Clustering Ants”, in *Proceedings Third Intl. Conf. on Simulation of Adaptive Behaviour: from Animals to Animats, Cambridge, MIT Press*, 1994, pp. 499-508.

[18] N. Monmarché, M. Slimane and G. Venturini, “AntClass: Discovery of Clusters in Numeric Data by a Hybridization of an Ant Colony with the K-Means Algorithm”. *Tech. Report 213, Information Laboratory*, University of Tours, France, 1999.

BIOGRAPHY

Weng Kin Lai obtained his MSc (Electronics) from the Queens University in Belfast (U.K.) and the PhD in Electrical and Electronics Engineering from the Auckland University. He is currently the Head of the research group in Machine Intelligence and Information Processing at MIMOS. His current research interests include machine intelligence, information processing, channel assignment for cellular networks, and biometrics.

Kok Meng Hoe is currently pursuing his M.Sc. (by Research) at the University of Science Malaysia (USM) in Penang, Malaysia. He obtained his BSc in Computer Science from USM. He is member of the research group in Machine Intelligence and Information Processing at MIMOS. His research interests are intelligent information processing, machine learning, data and text mining.

Yit Mei Aw holds a Degree of Bachelor of Science from Campbell University (U.S.A). She is a member of the research group in Cryptography at MIMOS. Her research interests are swarm intelligence and cryptography.

Tracy Sock Yin Tai holds a BSC (Hons) in Computing from the University of Portsmouth (U.K). She is a senior member of the research group in Machine Intelligence and Information Processing at MIMOS. Since the early part of 2002, she has been on study leave pursuing her Masters (Computer Science) at the University of Malaya. Her current research interests include information retrieval, text mining and agent technologies.