Available online at www.sciencedirect.com
SCIENCE DIRECT
Journal of Systems Architecture 52 (2006) 505–515
JOURNAL OF SYSTEMS ARCHITECTURE
www.elsevier.com/locate/sysarc

A flocking based algorithm for document clustering analysis

Xiaohui Cui *, Jinzhu Gao, Thomas E. Potok

Oak Ridge National Laboratory, Oak Ridge, TN 37831-6085, United States

Available online 19 April 2006

Abstract

Social animals or insects in nature often exhibit a form of emergent collective behavior known as *flocking*. In this paper, we present a novel Flocking based approach for document clustering analysis. Our Flocking clustering algorithm uses stochastic and heuristic principles discovered from observing bird flocks or fish schools. Unlike other partition clustering algorithm such as *K*-means, the Flocking based algorithm does not require initial partitional seeds. The algorithm generates a clustering of a given set of data through the embedding of the high-dimensional data items on a two-dimensional grid for easy clustering result retrieval and visualization. Inspired by the self-organized behavior of bird flocks, we represent each document object with a *flock boid*. The simple local rules followed by each *flock boid* result in the entire document flock generating complex global behaviors, which eventually result in a clustering of the documents. We evaluate the efficiency of our algorithm with both a synthetic dataset and a real document collection that includes 100 news articles collected from the Internet. Our results show that the Flocking clustering algorithm achieves better performance compared to the *K*-means and the Ant clustering algorithm for real document clustering.

Keywords: Document clustering; Bio-inspired; Agent; Flocking model; F-measure

1. Introduction

Ant colonies, bird flocks, and swarm of bees etc. are all considered as multi-agent based models that exhibit a collective behavior. New algorithms based on these biological models have been invented to solve problems in computer science. These algorithms are characterized by the interaction of a large number of agents that follow the same rules. The Flocking model is one of the first collective behavior models that have been applied in popular applications such as animation. Flocking is often considered as an Artificial Life algorithm because of its emergent property.

The Flocking model was first proposed by Craig Reynolds in his paper "Flocks, herd, and schools: A distributed behavior model" [24]. It is a bio-inspired computational model for simulating the animation of a flock of entities called *boids*. It represents group movement as seen in bird flocks and schools of fish in nature. In this model, each boid makes its own decisions on its movement according to a small number of simple rules that react to the neighboring mates in the flock and the environment it can sense. The simple local rules of each boid generate complex global behaviors of the entire flock. In addition of being used to simulate group motion which has

* Corresponding author. Tel.: +1 865 576 9654; fax: +1 865 576 0003.
E-mail addresses: cuix@ornl.gov (X. Cui), gaoj@ornl.gov (J. Gao), potokte@ornl.gov (T.E. Potok).

1383-7621/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.sysarc.2006.02.003

been used in a number of movies and games, Flocking behavior has already been used for time-varying data visualization [19,22] and for spatial cluster retrieval [9,17]. However, it appears that Flocking based algorithms have not been used to cluster a text document collection. In this study, a document clustering algorithm based on the Flocking behavior is proposed.

The remainder of this paper is organized as follows: Section 2 describes the related works in the document clustering area. Section 3 provides the methods of representing documents in clustering algorithms and of computing the similarity between documents. Section 4 provides a general overview of the Flocking model and introduction of the Flocking clustering algorithm. Section 5 provides the detailed experimental setup and results for comparing the performance of the Flocking clustering algorithm with the *K*-means and Ant clustering approaches. The discussion of the experiment's results is also presented. The conclusion is in Section 6.

2. Related works

Document clustering is a fundamental operation used in unsupervised document organization, automatic topic extraction, and information retrieval. It provides a structure for organizing large bodies of text for efficient browsing and searching [27]. Clustering involves dividing a set of objects into a number of clusters. The motivation behind clustering a set of data is to find inherent structure in the data and to expose this structure as a set of groups [1,3]. The data objects within each group should exhibit a large degree of similarity, while the similarity between different clusters should be minimized [2,13,29]. There are two major clustering techniques: "Partitioning" and "Hierarchical" [2,13]. Most document clustering algorithms can be classified into these two groups. Hierarchical techniques produce a nested sequence of partitioning, with a single, all-inclusive cluster at the top and a set of single clusters of individual points at the bottom. The partitioning clustering method seeks to partition a collection of documents into a set of non-overlapping groups, so as to maximize the evaluation value of clustering. Although the hierarchical clustering technique is often portrayed as a clustering approach with better quality, it does not contain any provision for the reallocation of entities, which may have been poorly classified in the early stages of the text analysis [13]. Moreover, the time complexity of this approach is quadratic [29].

In recent years, it has been recognized that the partitioning technique is well suited for clustering a large document dataset due to their relatively low computational requirements [29]. The time complexity of the partitioning technique is almost linear, which makes it widely used. The best-known partitioning algorithm is the *K*-means algorithm and its variants [12,28]. This algorithm is simple, straightforward and is based on the firm foundation of analysis of variances. The *K*-means algorithm clusters a group of data vectors into a predefined number of clusters. It starts with a random initial cluster center and keeps reassigning the data objects in the dataset to cluster centers based on the similarity between the data object and the cluster center. The reassignment procedure will not stop until a convergence criterion is met (e.g., the fixed iteration number or the cluster result does not change after a certain number of iterations). The main drawback of the *K*-means algorithm is that the cluster result is sensitive to the selection of the initial cluster centroids and may converge to the local optima [13]. Therefore, the initial selection of the cluster centroids decides the main processing of *K*-means and the partitioning result of the dataset as well. Another limitation of the *K*-means algorithm is that it generally requires a prior knowledge of the probable number of clusters for a data collection.

To deal with the limitations that exist in traditional partition clustering methods a number of computer scientists in recent years have proposed several approaches inspired from biological collective behaviors to solve the clustering problem, such as Genetic Algorithm (GA) [14], Particle Swarm Optimization (PSO) [4,5,18,20], Ant clustering [6,7,10,15] and Self-Organizing Maps (SOM) [30]. Within these clustering algorithms, Ant clustering algorithm is a partitioning algorithm that does not require a prior knowledge of the probable number to clusters or the initial partition. The Ant clustering algorithm was inspired by the clustering of corpses and eggs observed in the real ant colony. Deneubourg et al. [6] proposed a "Basic Model" to explain the ants' behavior of piling corpses and eggs. In their study, a population of ant-like agents randomly moved in a 2D grid. Each agent only follows one sample rule: randomly moving in the grid and establishing a probability of picking up the data object it meets if it is free of load or establishing a probability of dropping down the data object if it

is loading the data object. After several iterations, a clustering result emerges from the collective activities of these agents. Lumer, Faieta and other researchers [16] extended this "Basic Model" and applied it to numerical data analysis. Wu [31] and Handl [11] proposed the use of Ant clustering algorithms for document clustering and declared that the clustering results from their experiments are much better than that from *K*-means algorithm. However, in Ant clustering algorithm, the clustered data objects do not have mobility themselves. The data objects' movements have to be implemented through the movements of a small number of ant agents, which will slow down the clustering speed. Since each ant agent that is carrying an isolated data object does not communicate with other ant agents, it does not know the best location to drop the data object. The ant agent has to move or jump randomly in the grid space until it finds a place that satisfies its object dropping criteria, which usually consumes a large amount of computation time. Our experiments show that Ant clustering algorithm needs more iterations to generate an acceptable clustering result. In this paper, we present a novel Flocking based clustering approach for document clustering analysis. Like the Ant clustering algorithm, the Flocking algorithm is a partitioning algorithm and does not require a prior knowledge of the number to clusters in the datasets. It generates a clustering of a given set of data through the projecting of the high-dimensional data items on a two-dimensional grid for easy clustering result retrieval and visualization. However, the Flocking algorithm is more efficient than the Ant clustering algorithm because each document object in the collection is projected as an agent moving in the virtual space, and each agent's moving activity is heuristic as opposed to the random activity in Ant clustering algorithm. In the following section, we explain how the Flocking based algorithm is applied to document clustering applications.

3. Preliminaries

3.1. Document representation

In most clustering algorithms, the dataset to be clustered is represented as a set of vectors $X = \{x_1, x_2, \ldots, x_n\}$, where the vector x_i corresponds to a single document object and is called a "feature vector" that contains proper features to represent the object. The text document objects can then be represented using the Vector Space Model (VSM) [8]. In this model, the content of a document is formalized as a point in the multi-dimensional space represented by a vector x, such as $x = (w_1, w_2, \ldots, w_n)$, where w_i $(i = 1, 2, \ldots, n)$ is the term weight of the term t_i in one document. The term weight value w_i represents the significance of this term in a document. To calculate the term weight, the occurrence frequency of the term within a document and in the entire set of documents must be considered. The most widely used weighting scheme combines the Term Frequency with Inverse Document Frequency (TF-IDF) [8]. The weight of term i in document j is given in Eq. (1):

$$w_{ji} = tf_{ji} \times idf_{ji} = tf_{ji} \times \log_2(n/df_{ji}) \tag{1}$$

where tf_{ji} is the number of occurrences of term i in the document j; df_{ji} indicates the term frequency in the collections of documents; and n is the total number of documents in the collection. This weighting scheme discounts the frequent words with little discriminating power. A word with a high frequency within a document and low frequency within the document collection will be assigned a high weight value. Before translating the document collection into TF-IDF VSM, the very common words (e.g. function words: "a", "the", "in", "to"; pronouns: "I", "he", "she", "it") are stripped out completely and different forms of a word are reduced to one canonical form by using Porter's algorithm [21].

3.2. The similarity metric

The similarity between two documents needs to be measured in clustering analysis. In order to group similar data objects, a proximity metric has to be used to identify objects that are similar. Over the years, two prominent ways have been proposed to compute the similarity between documents x_p and x_j. The first method is based on Minkowski distances [3], given by

$$D_n(x_p, x_j) = \left(\sum_{k=1}^{d_x} |w_{k,p} - w_{k,j}|^n \right)^{1/n} \tag{2}$$

where x_p and x_j are two document vectors; dx denotes the dimension number of the vector space; $w_{k,p}$ and $m_{k,j}$ stands for the documents x_p and x_j's weight values in dimension k. Another commonly used similarity measurement in document clustering is the cosine correlation measure [3,13,26], given by

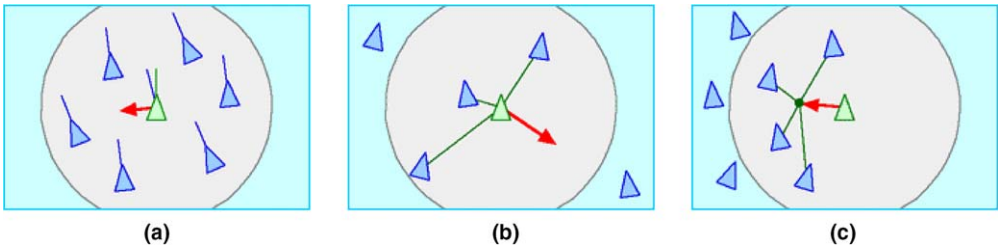

Fig. 1. The three basic rules in the boid [25]. (a) Alightment, (b) separation, (c) cohesion.

$$\cos(x_p, x_j) = \frac{x_p x_j}{|x_p||x_j|} \tag{3}$$

where $x_p x_j$ denotes the dot-product of the two document vectors and $|\cdot|$ indicates the length of the vector. Both similarity metrics are widely used in the text document clustering literature. In our proposed algorithm, we chose the Euclidean distance (Minkowski distances where $n = 2$) as the similarity metric. In order to manipulate equivalent threshold distances, considering that the distance ranges will vary according to the dimension number, this algorithm uses the normalized Euclidean distance as the similarity metric of two documents, x_p and x_j, in the vector space. Eq. (4) represents the distance measurement formula:

$$d(x_p, x_j) = \sqrt{\sum_{k=1}^{d_x} (w_{k,p} - w_{k,j})^2} \tag{4}$$

where x_p and x_j are two document vectors; dx denotes the dimension number of the vector space; $w_{k,p}$ and $m_{k,j}$ stand for the documents x_p and x_j's weight values in dimension k.

4. The flocking based algorithm

A basic Flocking model consists of three simple steering rules [25] that need to be executed at each instance over time, for each individual agent. These basic rules are listed below and shown in Fig. 1.

Rule 1: *Separation*, steering to avoid collision with other boids nearby.
Rule 2: *Alignment*, steering toward the average heading and speed of the neighboring flock mates.
Rule 3: *Cohesion*, steering to the average position of the neighboring flock mates.

In [25], a boid is used to represent the individual agent in the flock. In the circled area of Fig. 1(a), (b) and (c), the green triangle (located in the center of the circle) boid's behavior shows how a boid reacts to other characters in its local neighborhood.[1] The degree of locality is determined by the sensor range of the boid. The boid does not react to the flock mates outside its sensor range. Because a boid steers its movement based only on the local information, it has low complexity. These three basic rules of Reynolds's boids are sufficient to reproduce natural group behaviors on the computer.

The boids display the ability to adapt smoothly to unexpected external situations. However, these three basic rules will eventually result in all boids in the environment forming a single flock. It cannot reproduce the phenomena in the nature: the birds or other herd animals not only keep themselves within a flock that is composed of the same species or the same colony creatures, but also keep two different species or colony flocks separated. Inspired from the bird's ability to maintain a flock as well as separate different species or colony flocks, we believe that the Flocking model could offer a simple and heuristic way of clustering datasets. In our Flocking based document clustering algorithm, a fourth rule, feature similarity and dissimilarity rule, is added to influence the motion of the boids with the similarity among data objects.

For clustering the document collection, we assume each document vector is projected as a boid in a 2D virtual space. The document TFIDF vector is represented as the feature of the boid. Similar to birds in the real world, the boids that share similar document vector feature (same as bird's species and colony in nature) will automatically group together and become a boid flock. Other boids that have different document vector features will keep away from this flock. The behavior (velocity) of each boid B with position P_b is influenced by all boid X

[1] For interpretation of color in Figs. 1–4, the reader is referred to the web version of this article.

within position P_x in its neighborhood. It is driven by a set of local behavior rules, including the alignment rule, the cohesion rule, and the separation rule. The boid's velocity is also impacted by the feature similarity and dissimilarity compared to its nearby boids. These impacts react upon successive data updates, thereby generating distinct emergent motion typologies, which can be easily interpreted visually by human users. The three basic behavior rules and the feature similarity impacts can be illustrated by the following mathematical equations:

4.1. Alignment rule

The alignment rule, as shown in Fig. 1(a), acts as the active boid (outlined triangle located in the center of the diagram) trying to align its velocity vector with the average velocity vector of the flock in its local neighborhood. The degree of locality of this rule is determined by the sensor range of the active flock boid and is represented diagrammatically by the circle. The mathematical implementation is

$$d(P_x,P_b) \leqslant d_1 \cap d(P_x,P_b) \geqslant d_2 \Rightarrow \vec{v_{ar}} = \frac{1}{n}\sum_x^n \vec{v_x} \tag{5}$$

where v_{ar} is velocity driven by alignment rule, $d(p_x, p_b)$ is the distance between boid B and its neighbor X, n is the total number of boid B's local neighbors, v_x is the velocity of boid X, d_1 and d_2 are pre-defined distance values and $d_1 > d_2$.

4.2. Separation rule

The separation rule, as shown in Fig. 1(b), acts as an active boid trying to pull away before crashing into each other. The mathematical implementation is

$$d(P_x,P_b) \leqslant d_2 \Rightarrow \vec{v_{sr}} = \sum_x^n \frac{\overline{\vec{v_x}+\vec{v_b}}}{d(P_x,P_b)} \tag{6}$$

where v_{sr} is velocity driven by cohesion rule, d_2 is pre-defined distance, v_b and v_x are the velocities of boids B and X.

4.3. Cohesion rule

The cohesion rule, as shown in Fig. 1(c), acts as an active boid trying to orient its velocity vector in the direction of the centroid (average spatial position) of the local flock.

$$d(P_x,P_b) \leqslant d_1 \cap d(P_x,P_b) \geqslant d_2 \Rightarrow \vec{v_{cr}} = \sum_x^n (P_x - P_b) \tag{7}$$

where v_{cr} is velocity driven by cohesion rule, d_1 and d_2 are pre-defined distance and $(P_x \rightarrow P_b)$ calculates a directional vector point.

4.4. Feature similarity and dissimilarity rule

The flock boid tries to stay close to other boids that have similar features. For the document clustering algorithm, the boid's feature is represented by a document TFIDF vector. The strength of the attracting force is proportional to the distance between the boids and the similarity between the boids' feature values.

$$v_{ds} = \sum_x^n (S(B,X) \times d(P_x,P_b)) \tag{8}$$

where v_{ds} is the velocity driven by feature similarity, $S(B,X)$ is the similarity value between the features of boids B and X.

The flock boid tries to stay away from other boids that have dissimilar features. The strength of the repulsion force is inversely proportional to the distance between the boids and the similarity value between the boids' features.

$$v_{dd} = \sum_x^n \frac{1}{S(B,X) \times d(P_x,P_b)} \tag{9}$$

where v_{dd} is the velocity driven by feature dissimilarity. To achieve comprehensive Flocking behavior, the actions of all the rules are weighted and summed to give a net velocity vector required for the active flock boid.

$$v = w_{sr}v_{sr} + w_{ar}v_{ar} + w_{cr}v_{cr} + w_{ds}v_{ds} + w_{dd}v_{dd} \tag{10}$$

where v is the boid's velocity in the virtual space and $w_{sr}, w_{ar}, w_{cr}, w_{ds}, w_{dd}$ are pre-defined weight values.

5. Experiments and results

5.1. Experiment datasets

One synthetic dataset and one real document collection dataset are used for evaluating the performance of the clustering algorithms. The synthetic dataset consists of four data types, each including 200 two-dimensional (x,y) data objects. x and y are distributed according to Normal distribution

Table 1
The document collection dataset

	Category/topic	Number of articles
1	Airline safety	10
2	Amphetamine	10
3	China and Spy Plane and Captives	4
4	Hoof and Mouth Disease	9
5	Hurricane Katrina	5
6	Iran Nuclear	8
7	Korea and Nuclear Capability	10
8	Mortgage Rates	10
9	Ocean and Pollution	6
10	Saddam Hussein and WMD	8
11	Storm Irene	10
12	Volcano	10

$N(\mu, \sigma)$. This is the same dataset that has been used by Lumer and Faieta for their Ant clustering algorithm. There are many references in the document clustering literature [10,11,23] to use this synthetic dataset as a performance evaluation benchmark.

In the real document collection dataset, we used a document collection that contains 100 news articles. These articles are collected from the Internet at different time stages and have been categorized by human experts and manually clustered into 12 categories. A description of the test dataset is given in Table 1. In order to reduce the impact of the length variations of different documents, each document vector is normalized so that it is of unit length. Each term represents one dimension in the document vector space. The total number of terms in the 100 stripped test documents is 4790, which means the document collection has 4790 dimensions.

5.2. Experimental setup

The Flocking clustering algorithm, Ant clustering algorithm and *K*-means clustering algorithm are applied to the synthetic dataset and the real document collection dataset, respectively. The Euclidian distance measure is used as the similarity metric in each algorithm. No parameter needs to be set up for the *K*-means algorithm. However, it requires prior knowledge about how many clusters are expected in the dataset and the initial partition of the dataset.

For the Flocking clustering algorithm, each document is represented as one boid. All boids follow the four rules mentioned in Section four. The initial speed of each boid is set as $v_{\max} = 16$, because experimentally, this value rapidly generates clustering results that have good visualization for humans. Each boid can only sense the flock mates located within this sense range. The boid's sense range can be variables based on different virtual space sizes and boid numbers. The higher the sensor range, the faster the clustering result can emerge. But at the same time, each boid may need more computational resources to calculate its flying direction and speed at each iteration, especially, after several iterations, as the flock boids are grouped together. Each boid may have many neighboring flock mates needed to be considered if the sensor range is high. To balance quick results and high computational requirements, the flock boid's initial sensor range is set to a high value and gradually reduced to a low value after several iterations. The virtual grid space for document boids flying is set as a 500×500 2D square space. Each boid's velocity will gradually change base on the number of the flock mates in its surrounding area. The more flock mates in a boid, the slower the boid moves. To maintain the motion of the flock, a minimum speed $v_{\min} = 6$ is applied to all boids.

For the Ant clustering algorithm, our implementation modifies the Ant clustering code used in [11]. We modified the source codes so that we could cluster not only the synthetic dataset, but also the real text document dataset. Because of the extremely high dimensionality of the solution space of the text document dataset, 20 ant agents are used in the experiment instead of the 10 ant agents used in original code. In the Ant clustering algorithm implementation, for each iteration, the ant agent is randomly chosen to pickup or drop data. Altogether, there are 100 pickup and drop actions for the 20 agents per iteration, which equals to 100 flock boids' moving actions per iteration in the Flocking algorithm implementation. Other Ant algorithm parameters are kept same as the original source code.

5.3. Evaluation method

The results of the clustering algorithm should be evaluated using an informative quality measurement to reflect the quality of the clustering results. Depending on whether we have prior knowledge about the classification of the datasets, there are two kinds of evaluation methods. If there is no classification for the data, we have to use an "internal quality" measurement to evaluate the clustering results. However, there is no common agreement on the method of "internal quality" calculation.

The common internal quality measurement that we use in our partition clustering algorithm research [4] is the average similarity measurement.

Another evaluation method is measuring the clustering result and comparing it with the prior knowledge of the classification of the dataset. Since the document collection dataset used in our experiments has already been classified by a human expert, we will use the human classification results as a standard to evaluate these three clustering algorithms. We use the *F*-measure as the quality measure. This *F*-measure combines the precision and the recall ideas from information retrieve literature. The precision P and recall R of a cluster j (generated by the clustering algorithm) with respect to a class i (prior knowledge of the datasets) is defined as

$$P(i,j) = \frac{N_{ij}}{N_i} \tag{11}$$

$$R(i,j) = \frac{N_{ij}}{N_j} \tag{12}$$

where N_{ij} is the element number of class i within cluster j, N_j is the number of items of cluster j and N_i is the number of members of class i. The corresponding value of the *F*-measure is

$$F(i) = \frac{2PR}{P+R} \tag{13}$$

With respect to class i, members of the class i may be organized into different clusters, that will generate multiple *F*-measure value for class i. We consider the cluster with the highest *F*-measure score as the cluster for class i. The overall *F*-measure for the clustering result of one algorithm is computed as

$$F(i) = \frac{\sum_{i=1}^{n}(|i| \times F(i))}{\sum_{i}^{n}|i|} \tag{14}$$

where n is the number of the clusters in the dataset and $|i|$ is the number of data objects in class i. The *F*-value is limited within the interval $[0,1]$ with the higher the *F*-measure producing the better the clustering result.

5.4. Clustering results retrieve

The clustering results generated by the Ant and Flocking clustering algorithms are easily recognized by human eyes because of their visual character. However, it is necessary to quantify the clustering results so that they can be evaluated by the computer. The agglomerative hierarchical clustering method is used to retrieve the results. This method starts with a set of data objects as individual cluster, and then at each step the two most similar clusters merge. This process is repeated until the distance between all the clusters is larger than a specialized criteria. If individual data objects are not part of any of the clusters after the dataset are processed by Ant clustering algorithm or Flocking clustering algorithm, they are considered as outliers or noise. In our implementation of the Ant and Flocking clustering algorithms, these individual data objects are eliminated and are not counted as one cluster.

5.5. Experimental results

We evaluated the clustering methods over data sets representing distinct clustering difficulties in the same experimental conditions in order to better appreciate the performance of each clustering algorithm. The number of iterations in each algorithm was fixed at 300 iterations. First, we evaluated the *K*-means, Ant clustering and Flocking clustering over the synthetic dataset. Second, we tested the algorithms over the real document datasets. For each dataset, we ran each algorithm 20 times and computed the mean number of clusters found (since the *K*-means algorithm uses the prior knowledge of the cluster number of the data collection, the clustering number it produces is exactly equal to the real class number) and the *F*-measure of the clustering results. Table 2 shows the results obtained from both the synthetic and the real datasets. The three clustering algorithms all work well in the synthetic dataset. Figs. 2 and 3 show the visual results of the synthetic dataset clustered by the Ant and Flocking clustering algorithms. Four different colors (green, black, blue and red) are used to indicate different data classes. As shown in Figs. 2(a) and

Table 2
The results of *K*-means, Ant clustering and Flocking clustering Algorithm on synthetic and real datasets after 300 iterations

Document type	Algorithms	Average cluster number	Average *F*-measure value
Synthetic	Flocking	4	0.9997
Synthetic	*K*-means	(4)	0.9879
Synthetic	Ant	4	0.9823
Real	Flocking	10.083	0.8058
Real	*K*-means	(12)	0.6684
Real	Ant	1	0.1623

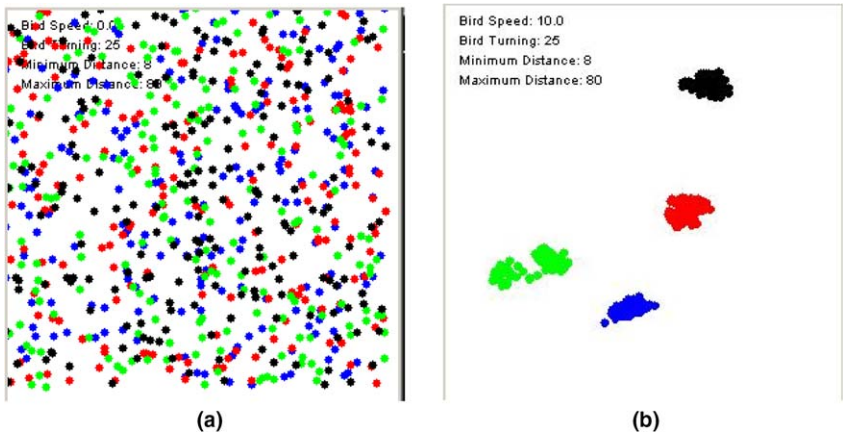

Fig. 2. The process of the Flocking clustering algorithm on 800 two-dimension synthetic data objects. (a) The initial data distribution in the space. (b) Data distribution after 300 iterations.

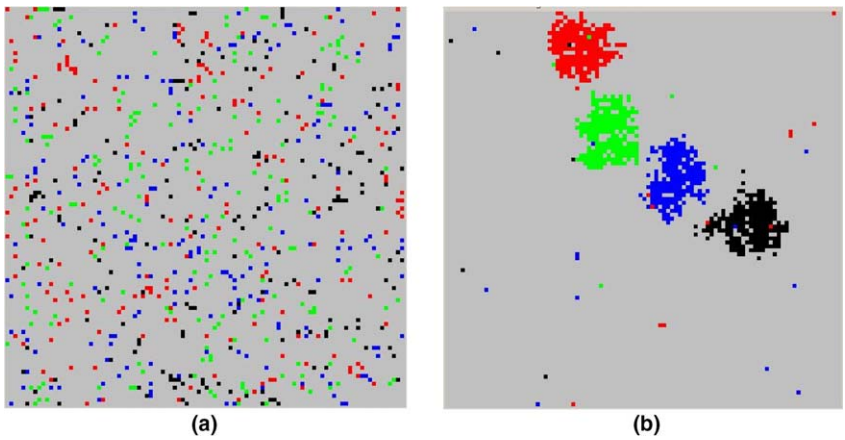

Fig. 3. The process of the Modified Ant clustering algorithm [11] on 800 two-dimension synthetic data objects. (a) The initial data distribution in the space. (b) Data distribution after 300 iterations.

3(a), the data objects belonging to different classes are randomly deployed in the space at the initial stage. After 300 iterations, based on visual observation, both the Ant and the Flocking algorithms generate good results while the Flocking clustering algorithm appears to generate the best clustering result.

When these three algorithms are applied to the 100 news article dataset, according to the results shown in Table 2, we determined that 300 iterations are not enough for the Ant clustering algorithm to generate an acceptable clustering result. But 300 iterations are sufficient for the Flocking clustering algorithm to generate good clustering results from the document dataset. Fig. 4 shows the process of the Flocking clustering algorithm that clusters the 100 articles dataset. The 100 articles are mirrored as 100 boids flying in a 2D space. The dataset has been manually sorted by human experts. Based on the class that the article belongs to, the boid is labeled by different colors for easy visualization. The twelve manually generated categories/topics are labeled in different colors and are displayed on the upper-left corner of the space as well. Each boid controls its own motion based on its neighboring flock mates' document vector similarity. After 300 iterations, shown in Fig. 4(b), the articles marked with the same color, which indicates that the articles belonging to same category are clustered together. One interesting phenomenon in Fig. 4(b) is two clusters circled in Fig. 4(b). These two clusters' contents do not match with the human sort result. According to the articles' labeled colors, one cluster includes Hurricane Katrina topic and Tropic Storm Irene

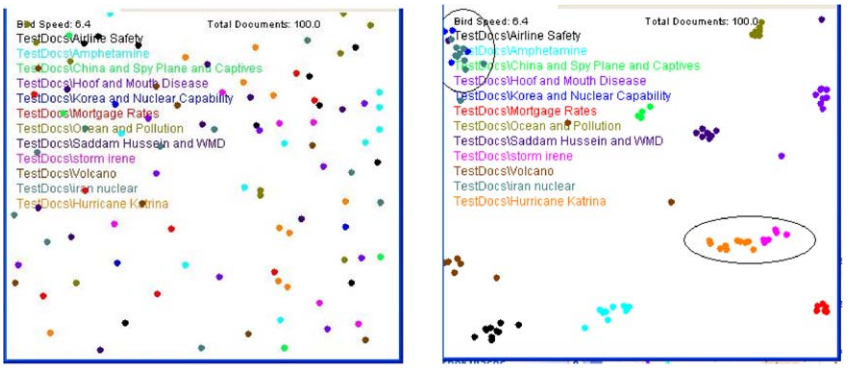

Fig. 4. Spatial distribution of 100 news articles on a 500×500 2D space processed by the Flocking clustering algorithm. (a) Initial articles distribution. (b) Articles distribution after 300 iterations.

topic. Another cluster includes Iran Nuclear topic and Korea and Nuclear Capability topic. We believe sorting these four topics into two topics, Storm topic and Nuclear Program, is a reasonable result, even though it differs from the manual classification.

Our experiment results also show that the *K*-means algorithm implementation need much less computing time and iterations to reach a stable clustering result than the other two algorithms. However, the drawback of the *K*-means clustering algorithm makes the average *F*-measure value of the clustering results lower than Flocking algorithm. The *K*-means algorithm also requires the probable number of clusters of a dataset before clustering it. For the Flocking clustering implementation and the Ant clustering implementation, the major computing time cost is the document vectors similarity and dissimilarity calculation part. In the present study, both implementations share the same similarity calculation code. Our experiment results show that it takes both implementations nearly same computing time to finish the initial 20–30 iterations. However, after that, the flocking implementation's computing time of each iteration quickly increases. This phenomenon can be explained as: in the Flocking implementation, the clustering result is generated very quickly and the boids with similar features quickly converge together, therefore, boids need to calculate the similarity values with multiple neighboring flock mates during the clustering result refining stage. For the Ant clustering algorithm implementation, our experiments show that even after thousands of iterations, the implementation still cannot generate an acceptable visual clustering result. The fact that, after several thousands of iterations, the computing time of each iteration is still keeping low may indicate most document objects are still randomly distributed in the grid space.

6. Conclusion

In this study, we present a new Flocking based document clustering algorithm. In this algorithm, each document in the dataset is represented by a boid. Each boid follows four simple local rules: the alignment rule, the separation rule, the cohesion rule, and the feature similarity and dissimilarity rule, to move in the virtual space. Boids following these simple local rules form complex and emergent global behaviors for the entire flock, and eventually these boids representing documents form a flock or cluster. Different flocks represent different document clusters. Similar to another bio-inspired clustering algorithm, the Ant clustering algorithm, and the Flocking algorithm does not need initial partitions or the prior knowledge about the class number for each dataset. The advantage of the Flocking clustering algorithm is the heuristic principle of the flock's searching mechanism. This heuristic searching mechanism helps boids quickly form a flock. Results from our experiments for evaluating these three different clustering algorithms illustrate that the Flocking clustering algorithm can generate a better clustering result with fewer iterations than that of the Ant clustering algorithm. The clustering results generated by the Flocking algorithm can be easily visualized and recognized by an untrained human user. Since the boid in the algorithm continues flying in the virtual space and joining the flock it belongs to, new results can be quickly re-generated when adding document boids or deleting part of

boids at run time. This feature allows the Flocking algorithm to be applied in clustering and analyzing dynamically changing information stream and real time visualizing of results for a human.

Acknowledgement

Oak Ridge National Laboratory is managed by UT-Battelle LLC for the US Department of Energy under contract number DE-AC05_00OR22725.

References

[1] M.R. Anderberg, Cluster Analysis for Applications, Academic Press, Inc., New York, NY, 1973.
[2] P. Berkhin, A survey of clustering data mining techniques, in: K. Jacob, N. Charles, T. Marc (Eds.), Grouping Multidimensional Data—Recent Advances in Clustering, Springer Publishers, 2006, pp. 25–72.
[3] K. Cios, W. Pedrycs, R. Swiniarski, Data Mining—Methods for Knowledge Discovery, Kluwer Academic Publishers, 1998.
[4] X. Cui, P. Palathingal, T.E. Potok, Document Clustering using Particle Swarm Optimization, in: IEEE Swarm Intelligence Symposium 2005, Pasadena, California, 2005, pp. 185–191.
[5] X. Cui, T.E. Potok, Document Clustering Analysis based on Hybrid PSO+*K*-means Algorithm, Journal of Computer Science, Special issue on Efficient heuristics for information organization, Science Publications, New York, 2005, pp. 27–33.
[6] J. Deneubourg, S. Goss, N. Franks, A. Sendova-Franks, C. Detrain, L. Chretien, The dynamics of collective sorting: Robot-like ants and ant-like robots, in: J.-A. Meyer, S. Wilson (Eds.), Proceedings of the First International Conference on Simulation of Adaptive Behaviour: From Animals to Animats 1, MIT Press, Cambridge, MA, 1991, pp. 356–365.
[7] M. Dorigo, E. Bonabeau, G. Theraulaz, Ant algorithms and stigmery, Future Generation Computer Systems 16 (8) (2000) 851–871.
[8] B. Everitt, Cluster Analysis, second ed., Halsted Press, New York, 1980.
[9] G. Folino, G. Spezzano, SPARROW: A Spatial Clustering Algorithm using Swarm Intelligence, Applied Informatics 2003 (AIA2003), Innsbruck, Austria, 2003, pp. 50–55.
[10] J. Handl, J. Knowles, M. Dorigo, Ant-based clustering: a comparative study of its relative performance with respect to *k*-means, average link and 1D-SOM. Technical Report TR/IRIDIA/2003–24. IRIDIA, Universite Libre de Bruxelles, Belgium, 2003.
[11] J. Handl, B. Meyer, Improved ant-based clustering and sorting in document retrieval interface, in: Proceedings of the Seventh International Conference on Parallel Problem Solving from Nature, LNCS, 2439, Springer-Verlag, Berlin, Germany, 2002, pp. 913–923.
[12] J.A. Hartigan, Clustering Algorithms, John Wiley and Sons, Inc., New York, NY, 1975.
[13] A.K. Jain, M.N. Murty, P.J. Flynn, Data clustering: a review, ACM Computing Survey 31 (3) (1999) 264–323.
[14] G. Jones, A. Robertson, C. Santimetvirul, P. Willett, Non-hierarchic document clustering using a genetic algorithm, Information Research 1 (1) (1995).
[15] N. Labroche, N. Monmarché, G. Venturini, AntClust: Ant Clustering and Web Usage Mining, Genetic and Evolutionary Computation Conference, 2003, pp. 25–36.
[16] E. Lumer, B. Faieta, Diversity and adaptation in populations of clustering ants, in: Proceedings of the Third International Conference on Simulation of Adaptive Behaviour: From Animals to Animats 3, MIT Press, Cambridge, MA, 1994, pp. 501–508.
[17] J. MacGill, S. Openshaw, The Use of Flocks to drive a Geographic Analysis Machine (online paper). Available from: <www.geocomputation.org>.
[18] V.D. Merwe, A.P. Engelbrecht, Data clustering using particle swarm optimization, in: Proceedings of IEEE Congress on Evolutionary Computation 2003 (CEC2003), Canbella, Australia, 2003, pp. 215–220.
[19] A.V. Moere, Information flocking: time-varying data visualization using Boid behaviors, in: Proceedings of the Eighth International Conference on Information Visualization, 2004, pp. 409–414.
[20] M. Omran, A. Salman, A.P. Engelbrecht, Image classification using particle swarm optimization, in: Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning 2002 (SEAL 2002), Singapore, 2002, pp. 370–374.
[21] M.F. Porter, An algorithm for suffix stripping, Program 14 (3) (1980) 130–137.
[22] G. Proctor, C. Winter, Information flocking: data visualisation in virtual worlds using emergent behaviours, in: Proceedings of the First international Conference on Virtual Worlds, in: J. Heudin (Ed.), Lecture Notes in Computer Science, vol. 1434, Springer-Verlag, London, 1998, pp. 168–176.
[23] V. Ramos, J. Merelo, Self-organized stigmergic document maps: environment as a mechanism for context learning, in: E. Alba, F. Herrera, J.J. Merelo, et al. (Eds.), 1st Spanish Conference on Evolutionary and Bio-inspired Algorithms, Merida, Spain, 2002, pp. 284–293, 6–8 Feb.
[24] C. Reynolds, Flocks, herds, and schools: a distributed behavioral model, Computer Graphics 21 (4) (1987) 25–34.
[25] C. Reynolds, Steering behaviors for autonomous characters, in: Proceedings of Game Developers Conference, San Jose, California, 1999, pp. 763–782.
[26] G. Salton, C. Buckley, Term-weighting approaches in automatic text retrieval, Information Processing and Management 24 (5) (1988) 513–523.
[27] G. Salton, Automatic Text Processing, Addison-Wesley, 1989.
[28] S.Z. Selim, M.A. Ismail, *K*-means type algorithms: a generalized convergence theorem and characterization of local optimality, IEEE Transactions on Pattern Analysis and Machine Intelligence 6 (1984) 81–87.
[29] M. Steinbach, G. Karypis, V. Kumar, A Comparison of Document Clustering Techniques, Text Mining Workshop, KDD, 2000.
[30] J. Vesanto, E. Alhoniemi, Clustering of the self-organizing map, IEEE Transactions on Neural Networks 11 (3) (2000) 586–600.
[31] B. Wu, Z. Shi, A clustering algorithm based on swarm intelligence, in: Info-tech and Info-net Proceedings, ICII 2001, vol. 3, 2001, pp. 58–66.

Xiaohui Cui received his M.S. degree in computer science from Wuhan University, China in July 2000, and his Ph.D. degree in Computer Science and Engineering from University of Louisville, USA in November 2004. He is currently a Postdoctoral Research Associate at the Applied Software Engineering Research Group in Oak Ridge National Laboratory. His research interests include Collective Intelligence of multi-agent system, Swarm Modeling, Data Mining and Knowledge Discovering, distributed computing, and sensor network. He is a member of IEEE computer society.

Jinzhu Gao received her B.S. and M.S. degree from Huazhong University of Science and Technology, China, and her Ph.D. degree in Computer Science and Engineering from The Ohio State University, USA in June 2004. She is currently a Postdoctoral Research Associate at the Network and Cluster Computing Group in Oak Ridge National Laboratory. Her research interests include scientific visualization, parallel and distributed computing, and computer graphics. She is a member of ACM and IEEE computer society.

Thomas E. Potok is the leader of the Applied Software Engineering Research Group at the Oak Ridge National Laboratory, where he manages a staff of 14 researchers. He is the principal investigator on a number intelligent software agent research projects. Prior to this he worked for 14 years at IBM's Software Solutions Laboratory in Research Triangle Park, North Carolina. Dr. Potok has a BS, MS, and Ph.D. in Computer Engineering all from North Carolina State University. He is an adjunct faculty member at the University of Tennessee, and a member of the ACM and IEEE Computer Society. He has authored numerous publications, has filed five software patents, and organized several workshops.