A Novel Ant Clustering Algorithm Based on Cellular Automata

Ling Chen[1,2] Xiaohua Xu[1] Yixin Chen[3] Ping He[1]
[1]Department of Computer Science, Yangzhou University, Yangzhou 225009, China
[2] National Key Lab of Novel Software Tech, Nanjing Univ., Nanjing 210093, China
[3]Department of Computer Science, Univ. of Illinois at Urbana-Champaign, Urbana, IL, 61801, U.S.A.
lchen@yzcn.net, ArterOne@hotmail.com, chen@manip.crhc.uiuc.edu

Abstract

Based on the principle of cellular automata in artificial life, an artificial Ants Sleeping Model (ASM) and an ant algorithm for cluster analysis (A^4C) are presented. Inspired by the behaviors of gregarious ant colonies, we use the ant agent to represent data object. In ASM, each ant has two states: sleeping state and active state. The ant's state is controlled by a function of the ant's fitness to the environment it locates and a probability for the ants becoming active. The state of an ant is determined only by its local information. By moving dynamically, the ants form different subgroups adaptively, and hence the data objects they represent are clustered. Experimental results show that the A^4C algorithm on ASM is significantly better than other clustering methods in terms of both speed and quality. It is adaptive, robust and efficient, achieving high autonomy, simplicity and efficiency.

Keywords:
Cellular automata, Swarm intelligence, Ant colony algorithm, Ants Sleeping Model

1. Introduction

Cellular Automata (CA), which was presented by J.von Neumann [1], can be used in the research on information theory and algorithms. CA is composed of identical and independent cells that are distributed in the cellular space and arranged regularly into a grid. Each cell has several limited states, and evolves in discrete time steps according to a set of rules. By the distributed computations on the grid, CA offers a new parallel method for some complex problems hard to partition. The concept of artificial ants is firstly proposed in CA to represent cells. Recently, inspired by the swarm intelligence shown through the social insects' self-organizing behavior, researchers created a new type of artificial ants to imitate the nature ants' behaviors, and named it artificial ant colony system, which is also a form of artificial life. Social insects have high swarm intelligence [2], [3]. Inspired by the social insects' (e.g. bees, ants) behaviors such as reproducing, foraging, nest building, brood sorting and territory defending, people have designed a series of algorithms that have been successfully applied to the areas of function optimization [3], combinational optimization [2], [4], network routing [5], robotics [6] and other scientific fields. Some researchers have achieved promising results in data mining by using the artificial ant colony. Deneubourg et al [7], [8] first proposed a Basic Model (BM) to explain the ants' behavior of piling corpses. Based on BM algorithm, Lumer and Faieta [9] presented a formula to measure the similarity between two data objects and designed the LF Algorithm for data clustering. BM and LF have become well-known models that have been extensively used in different applications. Kuntz et al [10] improved the LF algorithm and successfully applied it to graph partitioning and other related problems. Ramos et al [11] and Handl et al [12] recently applied the LF algorithm to text clustering and reported promising results.

However both BM and LF models separate ants from the being clustered data objects, which increases the amount of data arrays to be processed. Since these algorithms use much more parameters and information, they require large amount of memory space. Moreover, since the clustered data objects cannot move automatically and directly; the data movements have to be implemented indirectly through the ants' movements, which bring a large amount of extra information storage and computation burden because the ants make idle movement when carrying no data object. Moreover, since the ant carrying an isolated data object may never find a proper location to drop it, it may make an everlasting idle moving. It consumes large amount of computation time. To form a high-quality clustering, the time cost is even much higher.

According to Abraham Harold Maslow's hierarchy of needs theory [13], the security desire becomes more important for human beings when their primary physical needs are satisfied. By the same token, we notice that for such weak creatures like ants, they will naturally gather in groups with those who have similar features and repel those that are different. Even among a single group, there

Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'04)
0-7695-2101-0/04 $ 20.00 IEEE

are separate nests with intimate ants building nests next to each other while strangers building nests apart. We borrowed the principle of Cellular Automata in artificial life and proposed an Ants Sleeping Model (ASM) to explain the ants' behavior of searching for secure habitat. Based on ASM, we present an artificial ant algorithm of clustering (A^4C) in which each artificial ant is an intelligent agent representing an individual data object. We define a fitness function to measure the ants' similarity with their neighbors. Since each individual ant uses only a little local information to decide whether to be in active state or sleeping state, the whole ant group dynamically self-organizes into distinctive, independent subgroups within which highly similar ants are closely connected. The result of data objects clustering is therefore achieved. In ASM, we also presented several local movement strategies that speed up the clustering greatly. Experimental results show that, compared with BM and LF algorithm, algorithm A^4C based on ASM is much more direct and simpler for implementation. It is self-adaptive in adjusting parameters, has fewer restrictions on parameters, requires less computational cost, and has better clustering quality.

The rest of the paper is organized as follows. Section 2 presents the efficient ants sleeping model and the formal definition of ASM. Section 3 gives a novel ant-clustering algorithm based on ASM. Experimental results are shown in section 4 and section 5 concludes the paper.

2. Ants Sleeping Model

CA is a discrete method to solve partial differential equations. According to some transition rules, cells in CA transit their states in discrete time steps to simulate complex phenomena in physics, biology, ecology, geography, medicine and economics and other areas. Three distinct characters make CA very suitable in simulating complex phenomena. First, since the cells in CA can evolve simultaneously, it has great potential of parallelism. Second, since the transition of cells' states only depending on the local operations, no central control is required in CA. The last, just the simplest states transition can also give birth to complex behaviors.

Swarm intelligence refers to a pattern that simple individual behaviors lead to an exhibition of integral intelligence. For instance, the ability of the ant colony composed of simple individuals to accomplish some complex tasks is just a type of swarm intelligence. The character of self-organizing without central control is shared by both swarm intelligence and CA. In order to make full use of their advantages, we extend the classical CA model by combining it with the swarm intelligence, and present an Ants Sleeping Model (ASM) for clustering problem in data mining.

In their habitat, ants tend to group with those that have similar physiques. Even within an ant group, they like to have familiar fellows in the neighborhood. Due to the need for security, the ants are constantly choosing more comfortable and secure environment to sleep in. This is the inspiration for us to establish the artificial ants sleeping model (ASM).

In ASM, an ant has two states on a two-dimensional grid: active state and sleeping state. When the artificial ant's fitness is low, he has a higher probability to wake up and stay in active state. He will thus leave his original position to search for a more secure and comfortable position to sleep. When an ant locates a comfortable and secure position, he has a higher probability to sleep until the surrounding environment becomes less hospitable and actives him again. The major factors of ASM are described as follows.

Definition 1: Cellular Automata —A

CA is described by a 4-tuple $A = (S_d, Q, N, \delta)$. Here

- A represents a CA;
- S_d represents the space where cells are distributed, d is the dimension of the cellular space;
- Q is the set of cells' limited states;
- N represents the cellular neighborhood. $\forall z \in S_d,\ N(z) \subseteq S_d$;
- δ is a rule of states transition based on the neighborhood. According to the rule, cells can transit from one state to another. We have $\forall z \in S_d, f(z): Q^{|N(z)|} \mapsto Q$.

Based on the common formal definition mentioned above, CA can be extended and applied in artificial ants clustering.

Definition 2: The set of cellular states —agents' clustering information Q

Let Q represents the limited state set of cells. $Q = \{(i, c_i) \mid i \in [0..n] \wedge c_i \in [0..n]\}$. If $i \neq 0$, i represents the label of $agent_i$, c_i represents the label of the class which $agent_i$ belongs to. If $i = 0$, $c_i = 0$ means that there is no agent in the cell. Parameter n is the number of agents.

Definition 3: Cellular space — 2-D grid G

Let $G = [0..w(n)-1] \times [0..h(n)-1]$ represent the cellular space which is a 2-dimensional array of all positions (x, y), where $x \in [0..w(n)-1], y \in [0..h(n)-1], n \in Z^+, h(n) \in Z^+, w(n) \in Z^+$, $w(n)$ and $h(n)$ are functions related to n, the number of $agent$. We set

$$w(n) = 2(\lfloor \sqrt{n} \rfloor + 1),\ h(n) = 2(\lfloor \sqrt{n} \rfloor + 1) \qquad (2.1).$$

Each position on the grid is represented by a two-dimensional coordinate (*x, y*), and $G(x,y) \in Q$ represents the information at the position (*x, y*). If there is $agent_i$ on (x,y), $G(x,y)=(i,c_i)$. $G(x,y)=(0,0)$, when there is no agent on (x,y). We call the grid the agent's living environment and assume that the grid's upper bound is connected to the lower bound, and the left bound is connected to the right bound. While in the BM, the grid is a normal rectangle; the ASM uses a grid topologically equivalent to a spherical surface grid. The advantages of this grid are, on the one hand, it can ensure the equality of all the locations in the grid, which avoids the difference between center and bound, center and corner and lets the agent move freely on the grid unless blocked by other agents; on the other hand, the operation style of cellular automata can be borrowed to expedite the agents' movement and computation.

Definition 4: Cell — agent

Let an *agent* represent one data object, $agent_i$ represent the i^{th} *agent*, and *n* represent the number of the *agent*. The location of $agent_i$ is represented by (x_i, y_i), namely $G(agent_i)=G(x_i,y_i)=(i,\ c_i)$.

In ASM, each agent represents one data object. In the clustering algorithm, each agent representing one data object is closer to the nature of clustering problem. Because the agents (ants) can behave just as the birds of a feather flock together, while cannot categorize data objects.

Definition 5: Cellular neighborhood — agent's neighborhood N

$$N(agent_i)=\left\{(x \bmod w(n), y \bmod h(n)) \;\middle|\; |x-x_i|\le s_x,\ |y-y_i|\le s_y\right\} \tag{2.2}$$

$$L(agent_i)=L(x_i,y_i)=\left\{(x,y) \;\middle|\; (x,y)\in N(agent_i), G(x,y)=(0,0)\right\} \tag{2.3}$$

$N(agent_i)$ is called $agent_i$'s neighborhood, and $N(x_i,y_i)=N(agent_i)$,where $s_x \in Z^+, s_y \in Z^+ . s_x$ and s_y are the vision limits of $agent_i$'s in the horizontal and vertical direction. $L(agent_i)$ is used to denote a set of empty positions in $N(agent_i)$.

Definition 6: The measurement of distance — dissimilarity matrix D

Let $data_i=(z_{i1}, z_{i2}, \cdots, z_{ik}) \in R^k, k \in Z^+$, we define

$$d_{ij}=d(agent_i, agent_j)=d(data_i, data_j)=\left\|data_i-data_j\right\|_p \tag{2.4}$$

$$D=(d_{ij})_{n\times n}=(d(agent_i, agent_j))_{n\times n} \tag{2.5}$$

Because each agent represents one data object, $d(agent_i, agent_j)$ is determined by the distance between the data objects represented by $agent_i$ and $agent_j$ Normally, we take Euclidean distance where $p=2$.

Definition 7 : The measurement of fitness — f

We use $f(agent_i)$ to represent the current fitness of $agent_i$, which measures how well $agent_i$' fits into the current living environment.

$$f(agent_i)=\max\left\{0, \frac{1}{(2s_x+1)\times(2s_y+1)}\sum_{agent_j\in N(agent_i)}\left(1-\frac{d(agent_i,agent_j)}{\alpha_i}\right)\right\} \tag{2.6}$$

$$\alpha_i=\frac{1}{n-1}\sum_{j=1}^{n} d(agent_i, agent_j) \tag{2.7}$$

$$\alpha=\frac{1}{n\times(n-1)}\sum_{i=1}^{n}\sum_{j=1}^{n} d(agent_i, agent_j) \tag{2.8}$$

Here α_i represents the average distance between $agent_i$ and other agents, and is used to determine when $agent_i$ should deviate from other agents. Obviously, $f(agent)\in[0,1]$. Sometimes, we can use a constant α to substitute α_i, namely

$$f(agent_i)=\max\left\{0, \frac{1}{(2s_x+1)\times(2s_y+1)}\sum_{agent_j\in N(agent_i)}\left(1-\frac{d(agent_i,agent_j)}{\alpha}\right)\right\} \tag{2.9}$$

Definition 8 : The active probability — p_a

We use function $p_\alpha(agent_i)$ to denote the probability for $agent_i$ to be activated by the surrounding environment and enter active state. We define

$$p_a(agent_i)=\frac{\beta^\lambda}{\beta^\lambda+f(agent_i)^\lambda} \tag{2.10}$$

Here $\beta \in R^+$ is the threshold of agents' active fitness. Parameter $\lambda \in R^+$ is the active pressure of the agents, normally we take $\lambda=2$. It can easily be seen that when $f << \beta$, $p_\alpha(agent_i)$ is close to 1. This means if the fitness of $agent_i$ is much smaller than the threshold, $agent_i$ has a high probability to be activated and become active. The active $agent_i$ moves around in the grid, searching for a more comfortable place to sleep. When $f >> \beta$, $p_\alpha(agent_i)$ is close to 0. This means if the fitness

of $agent_i$ is much larger than the active threshold, $agent_i$ is unlikely to be waken up and will still sleep.

Definition 9 : The clustering rule — δ

We define δ as a set of the clustering rule which updates the class information of the agents. The following rules must be included in δ :

- If $agent_i$ is sleeping, its class label is the same as most of its neighbors'.
- If $agent_i$ is active, its class label is the same as its label.

Based on the definitions and explanations above, we will give a formal definition of ASM as follows.

Definition 10 : Ants Sleeping Model — ASM

ASM can be denoted by a 5-tuple ASM = (G, Q, N, δ, D). Here G represents a two-dimensional grid in which each position is a cell, Q represents the set of cells' limited states, δ is an updating rule for clustering information: $\forall agent_i \in G, \delta(agent_i) : Q^{|N(agent_i)|} \mapsto Q$. The next state of $agent_i$ is determined by δ through the interaction of $agent_i$ with other cells in the neighborhood. δ is also related to *agent's* activating probability p_a. $D = (d_{ij})_{n \times n}$ represents the dissimilarity matrix among agents.

Based on the definition 1 to 10, we have the following description of ASM. At the beginning of the algorithm, the agents are randomly scattered on the grid, not more than one agent per position. All the agents are in active state, randomly moving around on the grid following the moving strategy. The simplest moving strategy is to freely choose an unoccupied position in the neighborhood as the next destination.

When an agent moves to a new position, it will recalculate its current fitness *f* and probability p_a so as to decide whether it needs to continue moving. If the current p_a is small, the agent has a lower probability of continuing moving and higher probability of taking a rest at its current position. Otherwise the agent will stay in active state and continue moving. The agent's fitness is related to its heterogeneity with other agents in its neighborhood. When the agent feels insecure, it leaves its original position and searches for a comfortable position. When it finds a comfortable position, the agent will stop moving and take a rest. The agents influence each other's fitness during the movements. The influence is limited to the agents in neighborhood, and one agent can calculate its fitness and moving probability using only the local information. With increasing number of iterations, such movements gradually increase, making the agents that have less heterogeneity to gather together and those with more heterogeneity separate from each other. Eventually, similar agents are gathered within a small area and have identical class label while different types of agents are located in separated areas and have different class labels.

ASM is essentially equal to cellular automata in the sense that the local effect can expand to the whole living environment of the agents and cause some global effects. ASM is direct and simple for operation. Using only local information, the agents can update information and send it through the grid to other agents. Through the cooperative effect, the agents dynamically form into clusters.

3. Ant clustering algorithm based on ASM

Based on the ASM structures mentioned above, we design an Adaptive Artificial Ant Clustering Algorithm (A^4C). The framework of the algorithm is as follows.

Algorithm A^4C
//**A**daptive **A**rtificial **A**nt **C**lustering **A**lgorithm

1. initialized the parameters $\alpha, \beta, \lambda, \theta, t_{max}, t, s_x, s_y, k_\alpha, k_\lambda$
2. ***for*** each agent ***do***
3. place agent at randomly selected site on grid
4. ***end for***
5. ***while*** (not termination) //such as $t \le t_{max}$
6. ***for*** each agent ***do***
7. compute agent's fitness $f(agent)$ and activate probability $p_a(agent)$
8. $r \leftarrow random([0,1))$
9. ***if*** $r \le p_a$ ***then***
10. activate agent and move to random selected neighbor's site not occupied by other agents
11. ***else***
12. stay at current site and sleep
13. ***end if***
14. update agent's class label using the rule δ
15. ***end for***
16. adaptively update parameters α, λ, $t \leftarrow t+1$
17. ***end while***
18. ***output*** clustering information of agents

Table 1. High-level description of A^4C (adaptive artificial ant clustering algorithm)

Some details of the algorithm should be explained as the following.

3.1. The fitness of the agents

Line 7 of the algorithm computes the fitness of the agents according to (2.6) or (2.9). There are several approaches to determine the value of α in (2.9).

① The simplest way is to let α be a constant. For instance we can let the value of α be equal to the mean distance between all the agents (see (2.8)). Since the distance $\frac{d(agent_i, agent_j)}{\alpha}$ between $agent_i$ and $agent_j$ can be calculated beforehand, the value of α can also be computed before the procedure of clustering.

② Assign each agent with different α value. Denote the α value of $agent_i$ as α_i, we can let α_i be the mean distance from $agent_i$ to all the other agents (see (2.7)). Similarly, by a preprocessing beforehand, the value of α_i and the value of $\frac{d(agent_i, agent_j)}{\alpha_i}$ can be calculated before the procedure of clustering.

③ The value of α is adjusted adaptively during the process of clustering. We use $\bar{f}_t = \frac{1}{n}\sum_{i=1}^{n} f_t(agent_i)$ to denote the average fitness of the agents in the *t*-th iteration. To a certain extent, $\bar{f}_t$ indicates the quality of the clustering. In line 16, which updates the parameters, the value of α can be modified adaptively using $\bar{f}_t$:

$$\alpha(t) = \alpha(t-\Delta t) - k_\alpha(\bar{f}_t - \bar{f}_{t-\Delta t}) \qquad (3.1)$$

Here k_α is a constant.

3.2. The active probability of the agents

The active probability of the agents is computed in Line 7 using (2.10). In (2.10), we suggest $\beta \in [0.05, 0.2]$ and normally $\beta = 0.1$. The parameter λ is a pressure coefficient of $p_\alpha(agent)$. Normally, there are two methods to determine the parameter λ as follows:

① λ is a constant. We suggest $\lambda = 2$.

② Adjust the value of λ adaptively. Generally, the ants can form a rough clustering rudiment fast in the initial stage of clustering, while at the latter stage they requires rather long time to improve the clustering since the precision needs to be raised. Therefore the activating pressure coefficient λ tends to change decreasingly on the whole. In Line 16 of A^{4}C, which updates the parameters, we can adjust the value of λ adaptively by the following function.

$$\lambda(t) = 2 + \frac{k_\lambda}{\bar{f}_t}\lg\frac{t_{\max}}{t} \qquad (3.2)$$

Suppose $t_{\max} \propto 10^6$, $\lambda(1) = 2 + \frac{k_\lambda}{\bar{f}_t}\lg t_{\max} \approx 2 + \frac{6k_\lambda}{\bar{f}_1}$, $\lambda(t_{\max}) = 2 + \frac{k_\lambda}{\bar{f}_{t_{\max}}}\lg 1 = 2$. According to our experience, if $t \to \frac{t_{\max}}{10}$ and $\bar{f}_t \geq 0.5$, $k_\lambda \approx 1$. Additionally, when $k_\lambda = 0$, ② degenerates to ①.

3.3. The strategy of agent's movement

Line 10 of the algorithm moves the activated agents to a new location. The activated $agent_i$ first select an empty position from its neighbor $L(agent_i)$ as its next position and then moves to it. To determine $agent_i$'s next position, several methods can be used.

① The random method. $agent_i$ selects a location from $L(agent_i)$ randomly.

② The greedy method. Let a parameter $\theta \in [0,1]$ determine $agent_i$'s probability to select the most suitable location from $L(agent_i)$ as its destination. We denote it as θ-greedy selecting method. This method can make full use of the local information.

4. Experimental results

In this section, we not only show the test result on the ant-based clustering data benchmark which was introduced by Lumer and Faieta in [10] to compare our method with that of Lumer and Faieta, we but also show the test results on several real data sets.

4.1. Ant-based Clustering Data Benchmark

First we test a data set with four data types each of which consists of 200 two-dimensional data (*x, y*) as shown in Figure 1(a). Here *x* and *y* obey normal distribution $N(\mu, \sigma^2)$. The normal distributions of the four types of data (*x, y*) are $\left(N(0.2,0.1^2), N(0.2,0.1^2)\right)$, $\left(N(0.2,0.1^2), N(0.8,0.1^2)\right)$, $\left(N(0.8,0.1^2), N(0.2,0.1^2)\right)$ and $\left(N(0.8,0.1^2), N(0.8,0.1^2)\right)$ respectively. Agents that belong to different cluster are represented by different symbols: o, +, *, ×.

We test the same benchmark using our algorithm A^{4}C. Figure 1(a) shows the clustered data. Figure 1(b) shows the initial distribution of the agents in the grid. Figure 1(c) to (d) shows the agents' distribution after 10000 and 20000 iterations respectively. In the test the parameters are set as: $\beta = 0.1$, $s_x \leftarrow 1$, $s_y \leftarrow 1$, $\theta \leftarrow 0.90$, $k_\alpha = 0.50$, $k_\lambda = 1$,

$w(n) = h(n) = 58$. The initialized value of α is $\alpha = \frac{1}{n \times (n-1)} \sum_{i=1}^{n} \sum_{j=1}^{n} d(agent_i, agent_j) \approx 0.5486$, and $\lambda(t) = 2 + \frac{k_\lambda}{\bar{f}_t} \lg \frac{t_{\max}}{t}$. In each iteration, the value of α is updated adaptively by formula $\alpha(t) = \alpha(t-50) - k_\alpha(\bar{f}_t - \bar{f}_{t-50})$ and the agents select their destination using greedy method. Compared to the best results of the LF [9], LF costs much more computation time (1000000 iterations) than ours (20000 iterations).

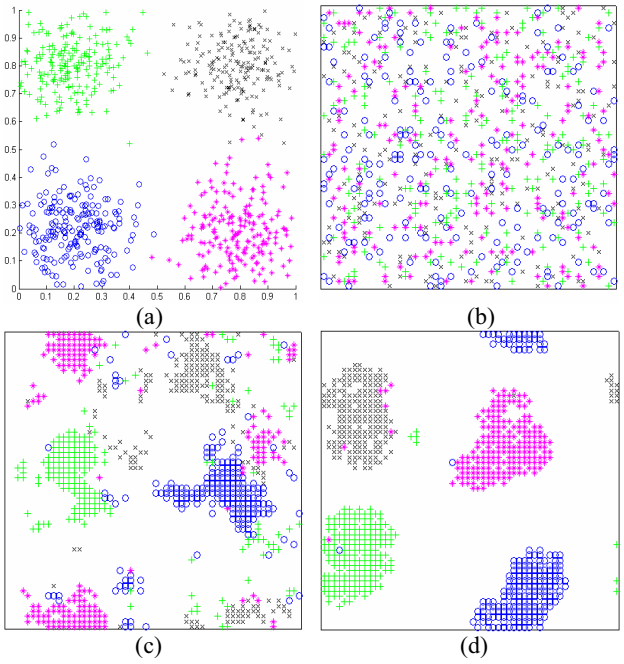

Figure 1. The process of clustering of A^{4}C.
(a) Data distribution in the attribute space
(b) Initial agent distribution in the grid
(c) Agents' distribution after 10000 iterations
(d) Agents' distribution after 20000 iterations

4.2. Real Data Sets Benchmarks

We test LF and A^{4}C using the real benchmarks of Iris and Glass. In the test, we consider two types of A^{4}C: one sets the parameters adaptively and we still denote it as A^{4}C, the other uses constant parameters and we call it SA^{4}C (special A^{4}C). The clustering results of A^{4}C after 5000 iterations are mostly better than those of LF after 1000000 iterations. We set the parameter t_{max} the maximum iterations of LF algorithm as 1000000, and those for A^{4}C and SA^{4}C as 5000. We set the parameters k_1=0.10, k_2=0.15 in LF and set θ=0.9, λ=2, k_α=0.5, k_λ=1, β=0.1 in SA^{4}C and A^{4}C. In the three algorithms, we set the size of the neighbor as 3×3. Results of 100 trials of each algorithm on the two data sets are shown in Table 2 to 3. It can easily be seen from the tables that SA^{4}C and A^{4}C require less iteration than LF. The quality of the clustering of SA^{4}C is better than LF, but is worse than A^{4}C.

	LF	SA^{4}C	A^{4}C
Maximum iterations t_{max}	**1000000**	5000	**5000**
Numbers of trials	100	100	100
Minimum errors	3	2	**2**
Maximum errors	13	8	**5**
Average errors	**6.68**	4.39	**2.13**
Percentage of the errors	4.45%	2.94%	**1.31%**
Average running time(s)	**56.81**	**1.36**	1.43

Table 2. Parameters and test results of LF, SA^{4}C and A^{4}C on Iris

	LF	SA^{4}C	A^{4}C
Maximum iterations t_{max}	**1000000**	5000	**5000**
Numbers of trials	100	100	100
Minimum errors	7	4	**2**
Maximum errors	12	12	**10**
Average errors	**10.25**	7.59	**3.94**
Percentage of the errors	4.79%	3.55%	**1.84%**
Average running time(s)	**106.21**	**2.37**	2.44

Table 3. Parameters and test results of LF, SA^{4}C and A^{4}C on Glass

Although the number of ants and data used in ASM is larger than that of LF, the ASM's overall computational cost is far less than LF's. Since in ASM many of the agents, especially those whose neighborhood are filled with ants, have already slept without moving, most computational resource is allocated to the poorly adapted ants, this helps to produce a fine clustering fast. Our test is completed on a PC (Pentium4 CPU 1.79GHz and

512M memory) with Matlab 6.0. The average speed of A^4C is much faster than that of LF.

From the test results above, it is obvious that the time cost of LF is quite large, mainly because ants spend a good deal of time in searching for data and have difficulty to drop its data in data-rich environment. Additionally, LF cannot deal with isolated points efficiently. Since it is very difficult for ants carrying an isolated data object to find a proper position to drop it down, they possibly make everlasting idle moving which consumes large amount of computational time. With referred to clustering quality, the parameters in LF algorithm, especially the parameter α, are difficult to set and can affect the quality of clustering results. Moreover, the parameters in LF lack adaptive adjustment of parameters which delays the process of clustering. Our A^4C algorithm based on ASM not only has advantages of simple, directness, dynamic, visible, also offer self-adaptive adjustment to important parameters and process isolated dada directly and effectively. Compared to LF algorithm, A^4C algorithm is more effective and can converge faster.

5. Conclusion

Inspired by the behaviors of gregarious ant colonies and according to the Maslow's hierarchy of needs theory, we extended Cellular Automata and presented a simple and novel Ants Sleeping Model (ASM), including its complete formal definition and explanation, to resolve clustering problems in data mining. In the ASM mode, each data is represented by an agent (artificial ant) with a small amount of basic information about the data per se and of its current position. In the agents' environment, which is a two-dimensional grid, each agent interacts with its neighbors and exerts influence on others. Those with similar features form into groups, and those with different features repel each other. Based on the ASM model, we proposed effective formulae for computing the fitness and activating probability of agents. We further analyzed the effect of each parameter, proposed several efficient agent moving strategies, and designed a self-adaptive ants clustering algorithm A^4C. In A^4C, the ants group can form into high-quality clusters by making simple moves according to little local neighborhood information.

Compared with BM and LF algorithm, the ASM and A^4C are direct, easy to implement, and self-adaptive. It has less parameter to tune, produces higher quality clusters, and is computationally much more efficient. We also proposed several ants moving strategies that have salient effect in fastening the clustering process. Experimental results on standard clustering benchmarks demonstrate significant improvement in both computation time and clustering quality by ASM and A^4C over previous methods.

References

[1] von Neumann J., *Theory of self reproducing cellular automata*. University of Illinois Press, Urbana and London, 1966.

[2] Bonabeau E., Dorigo M., Théraulaz G., *Swarm Intelligence: From Natural to Artificial Systems.* Santa Fe Institute in the Sciences of the Complexity, Oxford University Press, New York, Oxford, 1999.

[3] Kennedy J., Eberhart R.C., *Swarm Intelligence*. San Francisco, CA: Morgan Kaufmann Publishers. 2001.

[4] Dorigo M., Maniezzo V., Colorni A., *Ant system: optimization by a colony of cooperative learning approach to the traveling agents*[J]. IEEE Trans. On Systems, Man, and Cybernetics, 1996, 26(1):29-41

[5] Di Caro G., Dorigo M., *AntNet: A mobile agents approach for adaptive routing*, Technical Report, IRIDIA 97-12, 1997

[6] Holland O.E., Melhuish C., *Stigmergy, self-organization, and sorting in collective robotics,* Artificial Life 5,1999, pp.173~202.

[7] Dorigo M., Bonabeau E., Théraulaz G., *Ant Algorithms and stigmergy*, Future Generation Computer Systems 16(2000) 851~871.

[8] Deneubourg, J.-L., Goss, S., Franks, N., Sendova-Franks A., Detrain, C., Chretien, L. "*The Dynamic of Collective Sorting Robot-like Ants and Ant-like Robots*", SAB'90 - 1st Conf. On Simulation of Adaptive Behavior: From Animals to Animats, J.A. Meyer and S.W. Wilson (Eds.), 356-365. MIT Press, 1991.

[9] Lumer E., Faieta B., *Diversity and adaptation in populations of clustering ants*, in: J.-A.Meyer, S.W. Wilson(Eds.), Proceedings of the Third International Conference on Simulation of Adaptive Behavior: From Animats, Vol.3, MIT Press/Bradford Books,

[10] Kuntz P., Layzell P., Snyder D., *A colony of ant-like agents for partitioning in VLSI technology*, in: P. Husbands, I. Harvey(Eds.), Proceedings of the Fourth European Conference on Artificial Life, MIT Press, Cambridge, MA, 1997, pp.412~424.

[11] Ramos, Vitorino and Merelo, Juan J. "*Self-Organized Stigmergic Document Maps: Environment as a Mechanism for Context Learning*", in E.Alba, F. Herrera, J.J Merelo et al.(Eds.), AEB'02 – 1st Int. Conf. On Metaheuristics, Evolutionary and Bio-Inspired Algorithms, pp.284~293, Mérida, Spain, 6~8 Feb.2002.

[12] Handl, J. and Meyer, B. *Improved ant-based clustering and sorting in a document retrieval interface*, PPSN VII, LNCS 2439, 2002.

[13] http://www.maslow.org/