A stable clustering algorithm for mobile ad hoc networks based on attractor selection

Gen Nishikawa Fukuhito Ooshita Hirotsugu Kakugawa
Toshimitsu Masuzawa
Graduate School of Information Science and Technology, Osaka University
1-3 Machikaneyama, Toyonaka, Osaka, Japan
{g-nisikw | f-oosita | kakugawa | masuzawa}@ist.osaka-u.ac.jp

ABSTRACT

With the spread of wireless technology, mobile ad hoc networks is getting increased attention in recent years. In mobile ad hoc networks, network topologies dynamically change because of node mobility. Thus, it is important to design an algorithm that has strong stability against frequent topology changes. Attractor selection is one of the biologically inspired approaches that have strong stability against environmental changes. In this paper, we propose a stable clustering algorithm based on attractor selection for mobile ad hoc networks. Clustering is a fundamental problem for distributed systems and makes it easy to manage large scale networks. We show the effectiveness of our algorithm by simulations.

Keywords

Attractor selection, Distributed algorithm, Clustering, Mobile ad hoc networks, Stability

1. INTRODUCTION

Mobile ad hoc networks consist of a large number of mobile nodes that communicate with each other by wireless multi-hop communication. Mobile ad hoc networks can be established without any fixed infrastructure, and thus, they can be utilized various situation (event site, disaster site, etc). Then, mobile ad hoc networks are important technology to achieve ambient information networks.

In mobile ad hoc networks, network topologies change frequently since each node can move freely. Thus, an algorithm for mobile ad hoc networks should adaptively change a system state (i.e., local variables of each node). However, if an algorithm changes a system state every time topology changes, the system state is frequently changed. Since frequent changes of the system state cause heavy overhead, it is important to design an algorithm that has strong stability (an algorithm does not change the system state frequently) for topology changes in mobile ad hoc networks.

Attractor selection is a biological model first introduced by Kashiwagi et al. [2]. It models how E. coli cells adapt to changes in available nutrient even if molecular machinery for signal transduction to the DNA is unavailable. Attractor selection is a stochastic approach that decides a system state according to goodness of a current system state. If attractor selection evaluates the current system state as good, it keeps the current system states with high probability. On the other hand, if attractor selection evaluates the current system state as bad, it randomly changes the system state in order to find a new good state. By such a simple mechanism, attractor selection can find and keep a good state.

Leibnitz et al. proposed several algorithms based on attractor selection for routing in overlay networks [3, 4] and in mobile ad hoc networks [5]. However, to the best of our knowledge, there are few works on fundamental distributed algorithms based on attractor selection.

In this paper, we propose a clustering algorithm that can achieve stable clustering by adopting attractor selection. Clustering is a problem that divides a network into several groups called *clusters*. A cluster consists of one cluster head and several ordinary nodes. Clustering makes it easy to manage large scale networks and is helpful for mobile ad hoc networks. For example, clustering is used for hierarchical routing, that is, routing is divided into inter cluster routing and intra cluster routing. Each cluster head maintain information of the ordinary nodes in its cluster. Thus, frequent changes of cluster heads or ordinary nodes affect performance of routing (e.g., throughput, delay, etc).

There are many clustering algorithms for mobile ad hoc networks [1, 6, 7]. Johnen et al. proposed a clustering algorithm *RSCA* [1] for weighted networks where each node has a weight. A weight of a node represents suitability to become a cluster head (e.g. computing capacity, amount of the battery, bandwidth of wireless communication etc). *RSCA* satisfies the following *ad hoc clustering property*[8]:

Affiliation condition Every ordinary node p always affiliates with exactly one cluster head which has larger weight than p's weight

Cluster head condition For a given real number h, every ordinary node p whose cluster head $p.myhead$ has weight $(p.myhead).w$ and every cluster head q in p's neighbors, the weight of q is not larger than $(p.myhead).w + h$

k-neighborhood condition For a given integer k ($0 \leq k < n$), every cluster head has at most k neighboring cluster heads

Ad hoc clustering property adopts a fixed threshold to achieve stable clustering against network topology changes, however, unnecessary state changes still occur. For instance, in *RSCA*, if an ordinary node p whose cluster head has weight $(p.myhead).w$ temporarily has a neighboring cluster head q whose weight is larger than $(p.myhead).w + h$ and q leaves p's neighbors soon after, p always changes its cluster head and it may be unnecessary state

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Bionetics '08, November 25-28, 2008, Hyogo, Japan
Copyright 2008 ICST 978-963-9799-35-6.

changes. Our algorithm adopts the attractor selection scheme instead of a fixed threshold. Thus, in our algorithm, even if a node temporarily has a neighboring cluster head whose weight is larger than that of p's current cluster head, p does not always change its state. Therefore, our algorithm achieves strong stability.

The remaining of the paper is organized as follows. We describe the model we consider in this paper in Section 2. Then, we explain the original attractor selection model and Leibnitz's method in Section 3. In Section 4, we propose a clustering algorithm based on attractor selection for mobile ad hoc networks. Then, we show the simulation results and discuss them in Section 5. Finally, we present concluding remarks in Section 6.

2. PRELIMINARIES

A network is represented by an undirected graph $G = (V, E)$, where V is a set of nodes and E is a set of bidirectional communication links. Each node p has a unique identifier $id(p)$. For the sake of simplification, we refer to p instead of $id(p)$. We say nodes p and q are neighbors with each other if and only if link $(p, q) \in E$. We denote a set of neighbors of p as N_p. We assume that each node communicates by a *locally shared memory model*. That is, each node p has a finite set of local variables that can be read by nodes $q \in N_p \cup p$ and can only be updated by p. An algorithm of each node p consists of a set of *guarded commands* $\langle guard \rangle \rightarrow \langle action \rangle$. A $\langle guard \rangle$ represents a boolean expression that consists of the local variables of the node p and its neighbors N_p. An $\langle action \rangle$ represents an assignment that updates the local variables of the node p. A node is called *enabled* when it has a $\langle guard \rangle$ whose evaluation is *true*. These assumptions are same as those of *RSCA* [1].

Clustering is a problem to divide a network into several *clusters* such that each node belongs to exactly one cluster. In this paper, we consider clustering in mobile ad hoc networks where nodes can move freely. Thus, network topology changes arbitrarily and might be disconnected. Clustering in mobile ad hoc networks is defined as follows.

DEFINITION 1. *Clustering*
Let $\mathcal{G} = G_0, G_1, \cdots$ be the sequence of topologies of the target network where $G_t = (V, E_t)$ represents its topology at time t and E_{t+1} is obtained by removing some edges from E_t and adding some edges. Clustering *for $\mathcal{G}$ is to calculate the sequence $\mathcal{H} = H_0, H_1, \cdots$ where H_t is the set of cluster heads in G_t for $\forall t \geq 0$.*

In mobile ad hoc networks, it is necessary that each ordinary node can directly communicate with its cluster head. Thus, each node should satisfy the following *dominant property*.

DEFINITION 2. *Dominant property*
Every ordinary node has at least one cluster head in its neighbors.

We consider the weighted network where each node has a weight value which represents suitability to become a cluster head. A weight of a node is considered as its battery capacity or computing capacity, or bandwidth, etc. It is also preferred to minimize the number of clusters and changes of each node's state since a large number of clusters and frequently changes of each node's state cause enormous overhead. However, if one wants to achieve clustering with larger weight cluster head and a smaller number of clusters, nodes have to change their state more frequently. For instance, when two or more cluster head become neighbors due to mobility, the cluster head with smaller weight always changes its role to an ordinary node. Similarly, when an ordinary node p has a neighboring cluster head q whose weight is larger than that of p's current cluster head, p always changes its cluster head to q. That is, there is a trade-off between quality of clustering (weights of cluster heads and the number of clusters) and frequency of changes.

3. ATTRACTOR SELECTION

In this paper, we propose a biologically inspired clustering algorithm for mobile ad hoc networks. Our algorithm is based on attractor selection, which is proposed by Kashiwagi et al.[2]. Leibnitz et al. [3, 4, 5] extended the original 2-dimensional attractor selection to the M-dimensional one, and applied it to multi-path routing in overlay networks [3, 4] and in ad hoc networks [5].

Our algorithm is also based on Leibnitz's method. When a node finds a state that achieves better clustering than its current state, the node examines whether it changes its state or not by attractor selection. In this section, we explain the principles of attractor selection and Leibnitz's method.

3.1 Principles of attractor selection

In the original biological model [2], attractor selection considers the concentrations $m_i (i \in \{1, 2\})$ of two kinds of mRNA and activity α $(0 \leq \alpha \leq 1)$. Activity α represents goodness of current concentrations for current environment. The better current concentrations are, the larger activity α becomes. In E. coli cells, the concentrations of mRNA are adaptively changed into the good ones defined by activity. This behavior is represented as two differential equations:

$$\begin{aligned} \frac{dm_1}{dt} &= \frac{syn(\alpha)}{1 + m_2^2} - deg(\alpha) \times m_1 + \eta_1 \\ \frac{dm_2}{dt} &= \frac{syn(\alpha)}{1 + m_1^2} - deg(\alpha) \times m_2 + \eta_2 \end{aligned} \tag{1}$$

The term η_i is the white Gaussian noise. The functions $syn(\alpha)$ and $deg(\alpha)$ correspond to the rate coefficients of mRNA synthesis and degradation, respectively. They are both functions of activity α and are defined as follows:

$$syn(\alpha) = \frac{6\alpha}{2 + \alpha}, \qquad deg(\alpha) = \alpha. \tag{2}$$

The important character of expression (1) is that activity α changes the influence of a random term. If activity α is small the behaviors of concentrations of mRNA are influenced by the random term η_i and then concentrations of mRNA are randomly changed. On the other hand, if activity α is large the concentrations of mRNA are deterministically changed and converge to the values defined by expression (1).

3.2 Leibnitz's method

Leibnitz et al. [3, 4, 5] extended the 2-dimensional attractor selection to the M-dimensional case, and proposed the method that can adaptively select one good state from M states according to activity. Since Leibnitz's method inherits adaptability from the original attractor selection, it can find and keep a good state against the environmental changes. In Leibnitz's method, each m_i $(0 \leq i \leq M)$ represents that the i-th state is selected with probability $m_i / \sum_{k=1}^{M} m_k$. Activity α $(0 \leq \alpha \leq 1)$ represents goodness of the current value of each m_i $(0 \leq i \leq M)$. The dynamic behavior of each m_i is represented as M differential equations:

$$\frac{dm_i}{dt} = \frac{syn(\alpha)}{1 + m_{max}^2 - m_i^2} - deg(\alpha) \times m_i + \eta_i \qquad i = 1, \ldots, M, \tag{3}$$

where $m_{max} = \max\{m_i | 1 \leq i \leq M\}$. The functions $syn(\alpha)$ and

deg(α) are defined as follows:

$$\textit{syn}(\alpha) = \alpha \times (\beta\alpha^{\gamma} + \varphi^{*}), \qquad \textit{deg}(\alpha) = \alpha, \tag{4}$$

where the parameters β and γ are the factors which control the influence of activity to changes of m_i. In [5], Leibnitz's method adjusts $\beta = 50$ and $\gamma = 3$, and we use the same parameters in this paper. The value φ^{*} is a special offset which we describe in detail later. For simplicity, we define $\varphi(\alpha) = \textit{syn}(\alpha)/\textit{deg}(\alpha)$. Then, the equilibrium solution of expression (6) is the following:

$$m_i = \begin{cases} \varphi(\alpha) & m_i = m_{max}\,(H\text{-value}) \\ \frac{1}{2}\left[\sqrt{4+\varphi(\alpha)^2} - \varphi(\alpha)\right] & m_i \neq m_{max}\,(L\text{-value}). \end{cases} \tag{5}$$

From expression (5), Leibnitz's method selects the c-th state such that $m_c = m_{max}$ with high probability and rarely selects others. We call the c-th state such that $m_c = m_{max}$ as the inclined state. The value $\varphi(\alpha)$ needs to satisfy $\varphi(\alpha) \geq 1/\sqrt{2}$ in order that H-value becomes larger than or equal to L-value. Leibnitz's method adjusts $\varphi^{*} = 1/\sqrt{2}$, whereby H-value is equal to L-value when activity $\alpha = 0$.

The behavior of activity α is different in each method in [3], [4] and [5]. Thus, we explain only an overview here. In all methods, attractor selection knows the evaluation values of all states and calculates activity α to reflect the proportion of evaluation value of the inclined state to the best evaluation value among all states. When the inclined state is not good for the current environment, activity α becomes closer to 0. Then, the dynamic behavior of m_i is strongly dominated by random term η_i, and the difference between H-value and L-value becomes smaller. Therefore, the system randomly changes the inclined state and all states are selected with uniform probability. On the other hand, when the inclined state is good for the current environment, activity α becomes closer to 1. Then, the dynamic behavior of m_i is less dominated by random term η_i and the difference between H-value and L-value becomes larger. Therefore, the inclined state becomes stable and the system selects the inclined state with higher probability. By this mechanism, Leibnitz's method can find and keep a good state against the environmental changes.

4. CLUSTERING ALGORITHM

In this section, we show our clustering algorithm that achieves strong stability against frequent topology changes. Figure 1 shows our clustering algorithm that is based on *RSCA* [1]. In our algorithm, each node selects its state based on attractor selection instead of the fixed thresholds in *RSCA*. There are three kinds of roles for nodes (cluster head, ordinary node, and nearly ordinary node). A nearly ordinary node is a temporal role before resigning a cluster head. If a node p whose role is a cluster head wants to resign its role, p becomes a nearly ordinary node until there are no neighboring node that belongs to a cluster whose cluster head is p.

The local variable $p.Ch$ represents a role of a node p.

- If p is a cluster head $p.Ch = T$.
- If p is an ordinary node $p.Ch = F$.
- If p is a nearly ordinary node $p.Ch = NF$.

The local variable $p.myhead$ represents a cluster head that p is belonging to. If p is a cluster head or a nearly ordinary node, $p.myhead = p$.

The overview of our algorithm is that each node decides to be a cluster head or an ordinary node based on its weight and those of its neighbors. When a node p finds a candidate state that can achieve

Constants

$p.w : \mathbb{N}$ // the weight of node p

$c : \mathbb{R}$ // the constant value that controls the number of neighboring cluster heads

Local variable of node p

$p.Ch$: {T, F, NF} // T if node p is a cluster head, F if p is an ordinary node, and NF if p is a nearly ordinary node

$p.myhead$: *IDs* // the cluster head of node p

Macros

$p.NCh = \{q \in N_p : (q.Ch = T)\}$ // the set of cluster head in p's neighbors

$p.NCh^{+} = \{q \in N_p : (q.Ch = T) \wedge (q.w > p.w)\}$ // the set of cluster head in p's neighbors that has larger weight than that of p

$p.Cl = |p.NCh^{+}|$

Function

Boolean *select*(w_{curr}, w_{cand}) // *true* if attractor selection decides to change the state to *cand*, and *false* if attractor selection decides to keep current state *curr* (w_{cand} and w_{curr} are evaluation values of states *cand* and *curr*)

Guards

$G_1(p) = G_{11}(p) \vee G_{12}(p) \vee G_{13}(p)$

$G_{11}(p) \equiv (p.Ch \neq \text{T}) \wedge \{(p.NCh^{+} = \phi) \wedge (p.NCh \neq \phi) \wedge (\textit{select}(\textit{max}\{q.w : q \in p.NCh\}, p.w))$

$G_{12}(p) \equiv (p.Ch \neq \text{T}) \wedge (p.NCh = \phi)$

$G_{13}(p) \equiv (p.Ch = \text{T}) \wedge (p.myhead \neq p)$

$G_2(p) = G_{21}(p) \vee G_{22}(p) \vee G_{23}(p)$

$G_{21}(p) \equiv (p.Ch = \text{F}) \wedge (\exists q \in p.NCh : q.w > (p.myhead).w) \wedge (\textit{select}((p.myhead).w, \textit{max}\{q.w : q \in p.NCh\}))$

$G_{22}(p) \equiv (p.Ch = \text{F}) \wedge (p.myhead \notin p.NCh) \wedge \neg G_1(p)$

$G_{23}(p) \equiv (p.Ch = \text{NF}) \wedge (\forall q \in N_p : q.myhead \neq p) \wedge p.NCh \neq \phi$

$G_3(p) = G_{31}(p) \vee G_{32}(p)$

$G_{31}(p) \equiv (p.Ch = \text{T}) \wedge (p.Cl > 0) \wedge (\textit{select}(p.w, \textit{max}\{q.w : q \in p.NCh\} \times (1 + c \times p.Cl)))$

$G_{32}(p) \equiv (p.Ch = \text{NF}) \wedge (p.myhead \neq p)$

Actions

$A_1(p) : G_1(p) \rightarrow p.Ch := \text{T};\ p.myhead := p;$

$A_2(p) : G_2(p) \rightarrow p.Ch := \text{F};\ p.myhead := r : (r.w = \textit{max}\{q.w : q \in p.NCh\});$

$A_3(p) : G_3(p) \rightarrow p.Ch := \text{NF};\ p.myhead := p;$

Figure 1: Our clustering algorithm based on attractor selection

clustering with larger weight cluster heads or a smaller number of clusters than those of current clustering, p examines whether it changes its state or not by the function *select*. The function *select* decides its return value by attractor selection. That is, attractor selection evaluates goodness of p's current state based on evaluation values of p's current state and the candidate state. If attractor selection evaluates p's current state as good, the function *select* returns *false* with high probability. On the other hand, if attractor selection evaluates p's current state as bad, the function *select* returns *true* with high probability.

In the following section, we first explain the behavior of each node, and then explain the behavior of attractor selection (function *select*) in our algorithm.

4.1 Behavior of a node

In this section, we explain the behavior of a node p dividing into three cases (p is an ordinary node, a cluster head and a nearly ordinary node).

Behavior of an ordinary node.

When an ordinary node p does not belong to a cluster whose cluster head is not in N_p, p has to belong to one cluster whose cluster head is in N_p or become a cluster head in order to satisfy *dominant property*. We explain the behavior of a node p when p does not belong to any cluster in following three cases.

Case1: If p has some neighboring cluster heads whose weights are larger than $p.w$, p does not need to change its role to a cluster head. Thus, p joins the cluster whose cluster head has the highest weight in p's neighboring cluster heads (action A_2 is activated by G_{22}).

Case2: If all p's neighboring cluster heads have smaller weight than p's weight, p is the most suitable node to become a cluster head in its neighbors. However, p does not always have to change its role to a cluster head since there may be a cluster head whose weight is not too small compared with $p.w$. Thus, p examines whether it changes its role to a cluster head or not by the function *select* in G_{11} (if p decides to change its role, action A_1 is activated by G_{11}, otherwise action A_2 is activated by G_{22}). The evaluation values of p's current state and the candidate state (i.e., the state p becomes a cluster head) are $q.w$ (where q is the cluster head that has the highest weight in N_p) and $p.w$, respectively. That is, p is more likely to be a cluster head as the difference between p's weight and q's weight is larger.

Case3: If p does not have any neighboring cluster head, p always becomes a cluster head in order to satisfy *dominant property*. (action A_1 is activated by G_{12}).

When p belongs to a cluster and has a neighboring cluster head q whose weight is larger than $(p.\textit{myhead}).w$, q is a better cluster head to belong to than p's current cluster head. However, p does not always have to change its current cluster head since the weight of its current cluster head is not too small compared with $q.w$ Thus, p examines whether it changes its cluster to the cluster whose cluster head is q or not by function *select* in G_{21}. (if p decides to change its cluster, action A_2 is activated by G_{21}).

Behavior of a cluster head.

When a cluster head p has some neighboring cluster heads that have larger weight than that of p, p is not the most suitable node to be a cluster head in p's neighbors. Thus, p examines whether it changes its role to a nearly ordinary node or not by the function *select* in G_{31} (if p decides to change its role to a nearly ordinary node, action A_3 is activated by G_{31}). The evaluation values of p's current state and the candidate state (i.e., the state p changes its role to a nearly ordinary node) are decided based on the weights of p and a cluster head q that has the largest weight in p's neighboring cluster heads. In addition, the number of p's neighboring cluster heads is also considered, that is, the evaluation value of the candidate state becomes larger as p has a larger number of neighboring cluster heads. Thus, p is likely to resign as a cluster head when q has large weight and p has many neighboring cluster heads.

When p is a cluster head $p.\textit{myhead}$ must be equal to p. Thus, when $p.\textit{myhead}$ is not equal to p, $p.\textit{myhead}$ is updated to p (action A_3 is activated by G_{13}).

Behavior of a nearly ordinary node.

When p has no neighboring node whose cluster head is p, p does not need to be a nearly ordinary node any more. Thus, p changes its role to an ordinary node (action A_2 is activated by G_{23}).

When p is a nearly ordinary node $p.\textit{myhead}$ must be equal to p. Thus, when $p.\textit{myhead}$ is not equal to p, $p.\textit{myhead}$ is updated to p ((action A_3 is activated by G_{32}).

4.2 Attractor selection for clustering

In this section, we explain the behavior of attractor selection in our algorithm. In our algorithm, when a node finds a new state that can achieve better clustering (cluster head with larger weight or a small number of clusters), it examines whether it changes its state or not by attractor selection based on Leibnitz's method [3, 4, 5].

Selection manner.

The function $\textit{select}(w_{\textit{curr}}, w_{\textit{cand}})$ returns the boolean value that decides whether a node keeps current state *curr* or changes its state to a new state *cand*. The arguments $w_{\textit{curr}}$ and $w_{\textit{cand}}$ represent the evaluation values of the states *curr* and *cand*, respectively. The evaluation values reflect goodness of the states and are defined by the weights of nodes and the number of neighboring cluster heads. When a node finds a state whose evaluation value is larger than that of the current state (e.g. a node becomes neighbors with a cluster head that has larger weight than its current cluster head), it calculates the values $m_{\textit{curr}}$ and $m_{\textit{cand}}$ according to attractor selection. The values $m_{\textit{curr}}$ and $m_{\textit{cand}}$ decide the probability that the function $\textit{select}(w_{\textit{curr}}, w_{\textit{cand}})$ returns either *true* or *false*. In our algorithm, we define that the function $\textit{select}(w_{\textit{curr}}, w_{\textit{cand}})$ returns *true* with the probability $\frac{m_{\textit{cand}}^2}{m_{\textit{curr}}^2+m_{\textit{cand}}^2}$. That is, as $m_{\textit{cand}}$ is larger, a node changes its state with higher probability. For simplicity, we define $\max\{\textit{curr}, \textit{cand}\}$ as the state which attractor selection is likely to select. That is, $\max\{\textit{curr}, \textit{cand}\} = \textit{curr}$ when $m_{\textit{curr}}$ is larger than $m_{\textit{cand}}$, otherwise $\max\{\textit{curr}, \textit{cand}\} = \textit{cand}$.

The values $m_{\textit{curr}}$ and $m_{\textit{cand}}$ change according to their values and activity α. Activity α, which is described hereinafter in detail, represents goodness of the state $\max\{\textit{curr}, \textit{cand}\}$. The changes of $m_{\textit{curr}}$ and $m_{\textit{cand}}$ are defined as follows:

$$\frac{dm_{\textit{curr}}}{dt} = \frac{\textit{syn}(\alpha)}{1+(m_{\textit{max}})^2-(m_{\textit{curr}})^2} - \textit{deg}(\alpha) \times m_{\textit{curr}} + \eta_{\textit{curr}}$$
$$\frac{dm_{\textit{cand}}}{dt} = \frac{\textit{syn}(\alpha)}{1+(m_{\textit{max}})^2-(m_{\textit{cand}})^2} - \textit{deg}(\alpha) \times m_{\textit{cand}} + \eta_{\textit{cand}} \quad (6)$$

where $m_{\textit{max}} = \max\{m_{\textit{curr}}, m_{\textit{cand}}\}$. We define the functions $\textit{syn}(\alpha)$ and $\textit{deg}(\alpha)$ as the same ones in expression (4). As described in Section 3, a node selects the state $\max\{\textit{curr}, \textit{cand}\}$ with high probability.

Table 1: Parameters for our algorithm and *RSCA*

Our algorithm	s	0.5, 0.75, 1, 2
	δ	0.1
	c	0.2
RSCA [1]	k	0, 1, 2, 3
	h	5

Table 2: Evaluation criteria

Criteria	Detail
Number of clusters	The average number of clusters in the whole simulations
Average weight	The average weight of cluster heads each node p belongs to (i.e., $\sum_{\forall p \in V}(p.\mathit{myhead}).w/\|V\|$)
Number of elections	The number of times that each node changes its state from an ordinary node to a cluster head
Number of re-affiliations	The number of times that each node leaves its current cluster and joins another cluster

Behavior of activity.

Attractor selection selects the state $\max\{\mathit{curr}, \mathit{cand}\}$ with high probability. Thus, we should reflect goodness of the state $\max\{\mathit{curr}, \mathit{cand}\}$ to activity α. In our algorithm, the value m_{curr} is always larger than m_{cand} initially in order to evaluate goodness of the state *curr* compared with the state *cand*. We define the behavior of activity α as follows.

$$\frac{d\alpha}{dt} = \delta \times \left(\left(\frac{w_{\mathit{max}}}{w_{\mathit{cand}}} \right)^s - \alpha \right), \tag{7}$$

where w_{max} is a evaluation value of the state $\max\{\mathit{curr}, \mathit{cand}\}$. Expression (7) shows that activity α converges to $\left(\frac{w_{\mathit{max}}}{w_{\mathit{cand}}}\right)^s$. Thus, if the difference of evaluation values between states *curr* and *cand* is small, activity α is enough large and a node may keep its state to *curr*. On the other hand, if the difference of evaluation values between states *curr* and *cand* is large, activity α becomes small and a node may changes its state to *cand*. Note that, w_{cand} is always larger than w_{curr}. The parameters δ and s reflect the sensitivity of changing α. If δ is large, activity α rapidly responds to the environmental changes. If s is large, activity α becomes sensitive to the difference between w_{curr} and w_{cand}, and then, a node is likely to change its state to the state *cand*. Thus, the larger s is, the more our algorithm performs to achieve better clustering.

5. SIMULATIONS

In this section, we show simulation results that our algorithm achieves stable clustering. We compare our algorithm with *RSCA* [1]. The simulation results show that our algorithm can construct clustering whose quality is comparable to *RSCA* with less frequent state changes of nodes than those of *RSCA*. We first show the simulation settings and then the simulation results.

5.1 Simulation settings

We randomly deploy 200 nodes on a 500 $\times$ 500 square field. Each node has uniform communication radius 50 and random weight $[50.0..100.0]$. We performed 100 simulation experiments with different initial node deployment.

Each node moves according to the random way point model. In our simulations, each node selects a random destination and a random speed $[1.0..3.0]$, then it moves toward the destination with decided speed. After the node arrives at its destination, the node decides a random time $[0..6.0]$ and waits during the decided time. Then, the node repeats above actions until the end of simulation.

In each round, every node repeats executing the algorithm until there are no enable node (i.e., clustering in the current topology is completed). The calculation of attractor selection expressions is only once in every round in order that the return value of the function *select* is unchanged in the same round.

Other parameters in our algorithm and *RSCA* are shown in Table1. The parameters s and δ effect the sensitivity of changing α, and the parameter c effects the sensitivity of reducing neighboring cluster heads. The parameters k and h have the same meaning as those used in *Ad hoc clustering property* [8]. In *RSCA*, a cluster head has at most k neighboring cluster heads, and an ordinary node p has no neighboring cluster head q such that $q.w \geq (p.\mathit{myhead}).w + h$. We also performed simulation experiments with changing the value of h from 0 to 20, however, the results of *RSCA* are less affected by the value of h, and we omit these results due to the space limitation.

Table 2 shows the evaluation criteria in our simulation. Number of clusters and average weight reflect the quality of clusters. Number of elections and re-affiliations reflect the stability of a clustering algorithm.

We compare out algorithm to *RSCA* in terms of the stability when the quality of clusters are similar. Then, we can show our algorithm achieve more stable clustering than *RSCA*.

5.2 Simulation results

Figures 2 and 3 show the quality of clusters that each algorithm constructs. Figures 4 and 5 show the stability of each algorithm. Those results show that the difference between both algorithms is quite small for number of clusters and average weight. On the other hand, our algorithm has a smaller number of elections and re-affiliations than those of *RSCA*. Therefore, we can conclude that our algorithm achieves stronger stability than *RSCA*.

In this paragraph, we show the impact of parameters. When s becomes smaller in our algorithm or k becomes larger in *RSCA*, larger number of cluster heads becomes neighbors, and thus the number of cluster heads is increased as shown in Figure 2. When the number of clusters becomes large, the difference of the stability between our algorithm and *RSCA* becomes small as shown in Figures 4 and 5. The reason is that, when the number of clusters becomes large, most of cluster heads does not need to resign their role, and thus, the number of elections is decreased. In addition, the large number of clusters causes the small number of ordinary nodes, and thus, the number of re-affiliations is decreased. Thus, our algorithm achieves efficient clustering with strong stability, especially the small number of clusters are needed.

We also performed simulation experiments where maximum speed of nodes is 1 (lower mobility) and 5 (higher mobility). However, those results show the similar tendency when maximum speed of nodes is 3, and we omit those results due to the space limitation.

6. CONCLUSION

In this paper, we have proposed a clustering algorithm for mobile ad hoc networks based on attractor selection. Attractor selection decides whether it changes current system state or not according to goodness of current state. By applying attractor selection, our

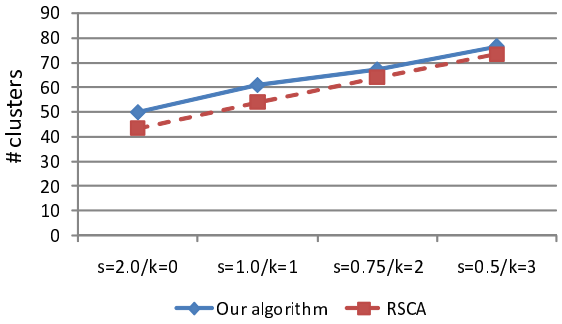

Figure 2: Number of clusters

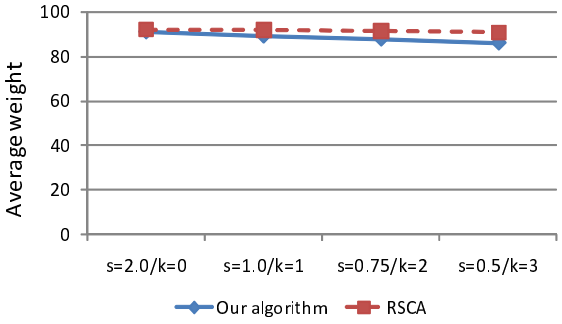

Figure 3: Average weight

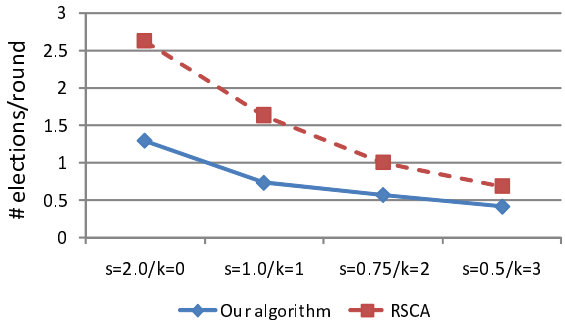

Figure 4: Number of elections

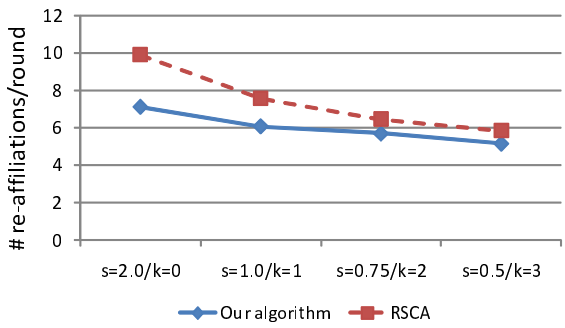

Figure 5: Number of re-affiliations

algorithm achieves strong stability against topology changes. In our algorithm, when a node finds a state that can achieve better clustering, it examines whether it changes the current state or not by attractor selection. We have shown by simulation experiments that our algorithm achieves better stability than *RSCA* [1].

Our algorithm is one example that adopts attractor selection to distributed algorithms in mobile ad hoc networks. We believe that attractor selection can be applied to many other distributed algorithms in mobile ad hoc networks.

Acknowledgment

This work is supported in part by “Special Coordination Funds for Promoting Science and Technology: Yuragi Project" of the Ministry of Education, Culture, Sports, Science and Technology, Japan, “Global COE (Centers of Excellence) Program" of the Ministry of Education, Culture, Sports, Science and Technology, Japan, Grant-in-Aid for Scientific Research ((B)17300020, (B)19300017, (B)20 300012) of JSPS, Grant-in-Aid for Young Scientists ((B)18700059) of JSPS, and the Kayamori Foundation of Informational Science Advancement.

7. REFERENCES

[1] C. Johnen and L. H. Nguyen. Robust self-stabilizing clustering algorithm. In Proceedings of OPODIS 2006, pp. 410–424, 2006.

[2] A. Kashiwagi, I. Urabe, K. Kaneko, and T. Yomo. Adaptive response of a gene network to environmental changes by fitness-induced attractor selection. *PLoS ONE*, 1(1), December 2006.

[3] K. Leibnitz, N. Wakamiya, and M. Murata. Resilient multi-path routing based on a biological attractor-selection scheme. In *Proceedings of the 2nd International Workshop on Biologically Inspired Approaches to Advanced Information Technology (Bio-ADIT 2006)*, January 2006.

[4] K. Leibnitz, N. Wakamiya, and M. Murata. Biologically inspired adaptive multi-path routing in overlay networks. In *Proceedings of IFIP/IEEE International Workshop on Self-Managed Systems and Services (SelfMan 2005)*, May 2005.

[5] K. Leibnitz, N. Wakamiya, and M. Murata. Self-adaptive ad-hoc/sensor network routing with attractor-selection. In *Proceedings of the 49th annual IEEE Global Telecommunications Conference (GLOBECOM 2006)*, November 2006.

[6] N. Mitton, A. Busson and E. Fleury. Self-organization in large scale ad hoc networks. In *Proceedings of the 3rd Mediterranean Ad-Hoc Networks*, 2004.

[7] N. Mitton, E. Fleury, I. G. Lassous, and S. Tixeuil. Self-stabilization in self-organized multihop wireless networks. In *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems Workshops*, 2005.

[8] S. Basagni. Distributed and mobility-adaptive clustering for multimedia support in multi-hop wireless networks. In *Proceedings of VTC 1999, pp. 889–893*, 1999.