J Supercomput (2012) 61:869–893
DOI 10.1007/s11227-011-0642-0

General criteria-based clustering method for multi-node computing system

Yu Niu · Brian J. d'Auriol · Sungyoung Lee

Published online: 5 July 2011

Abstract Synchronization/desynchronization and clustering are important techniques in multi-node computing systems, especially for sensor networks (SN) which is broadly considered to be a type of multi-node computing environment. However, most of the existing algorithms' clustering criteria are limited to the node location information and ignore the nature and characteristics of the nodes as well as the requirements of the applications. In this paper, an autonomic concurrent General Criteria-based Clustering (GCC) method is proposed for multi-node computing systems. The GCC method is based on the neuron oscillator pulse-coupling model and its clustering criteria can come from any node-related data or properties. The cluster member nodes share similar physical or logical properties and represent those relationships in the form of Logical Clusters (LCs). Due to the neuron dynamic system basis of the method, there is concurrency that exists both on the whole network and on each individual node. The simulation shows that the GCC method can generate diverse logical clusters and synchronization/desynchronization coexistence results with acceptable time and energy usage.

Keywords Clustering · Pulse-coupling oscillator · Multi-node computing system

1 Introduction

Multi-node computing in general refers to a wide range of distributed systems comprised of autonomous computing nodes communicating via some network. It includes

Y. Niu · B.J. d'Auriol · S. Lee (✉)
Dept. of Computer Engineering, Kyung Hee University, Yongin, South Korea
e-mail: sylee@oslab.khu.ac.kr

Y. Niu
e-mail: niuyu@oslab.khu.ac.kr

B.J. d'Auriol
e-mail: dauriol@oslab.khu.ac.kr

the network computing/ cloud computing, parallel computing, distributed computing, mobile ad hoc systems, etc. It may also include sensor networks especially in cases where the sensor nodes provide limited computational capabilities such as data aggregation and data clustering. In many cases, synchronization/desynchronization and clustering are useful for multi-node applications among the important aspects.

In this paper, we emphasize these techniques in sensor networks (SN) which we consider broadly to be a multi-node computing environment: such consists of multiple spatially distributed autonomous sensor nodes often with a limited processing unit where all nodes interact locally and some nodes interact with gateways-termed base stations, the latter often having more computational energy and communication resources. Synchronization and desynchronization are two basic primitives [1] for such multi-node sensor environments. Synchronization usually implies time synchronization and provides the time baseline for all the nodes and events in the network. Desynchronization makes the nodes active in different time intervals to avoid transmission collision or balance the working load more evenly amongst group nodes [2–4]. In addition, node clustering, which aims to increase scalability and reduce the complexity of network management, is common in sensor networks. However, most of the existing clustering algorithms group the nodes by applying some simple criteria, such as node location and communication costs, etc. These often used clustering criteria ignore the nature and characteristics of the sensor nodes as well as the requirements of the application level. Moreover, many existing algorithms require additional assistant techniques for communication in cluster networks. For example, some need a central node for controlling or special coding methods (Direct-Sequence Spread Spectrum (DSSS) in LEACH [5]) to avoid the collision among the nodes and clusters. And some others need additional time synchronization algorithms for data aggregation or other data recording functions.

In this paper, we propose a General Criteria-based Clustering (GCC) method which is generally suitable for multi-node computing environments, in particular wireless environments, and emphasize its application for sensor networks. The GCC method is based on neuron oscillators' interactive coupling (i.e. pulse-coupling oscillators) whereby the activity is imitated to produce independent clusters based on the oscillators' initial phase distribution. Because the phase is an abstract concept derived from mathematical system, practically its initial value can be mapped from any node-related data or properties: which means that any node-related characteristic data can be used as the clustering criteria. The GCC method consists of two levels of concurrency: the upper-level derived from a neuron dynamic system consists of all of the computing nodes in the environment that the nodes self-autonomously cooperate via pulse-coupling communications to effect a global clustering; the lower-level is specific to each node and consists of asynchronous node-to-node communication support, asynchronous and shared memory internal communication.

The GCC overcomes the limitations in the existing clustering techniques by considering the nature and characteristics of computing nodes. In particular, application level requirements can be used to influence the results of clustering. The use of the application level requirements leads to a connection between applications and clustering. On demand and dynamic changes in the applications may also be reflected in re-clustering. We believe that this is the first method to perform clustering based on general application information. Besides, our method allows capturing the logical relationships that exist in distributed computing nodes and representing those relationships in the form of Logical Clusters (LCs). In addition to the clustering, the GCC method also produces an advantageous result that the nodes within a cluster are phase-synchronized and phase-desynchronized among different clusters. If needed, these synchronized/desynchronized phases can also be reversely mapped as the sensor nodes' logical clock, which enables applications that need time synchronization or desynchronization without requiring any additional algorithms or procedures.

The paper is organized as follows. Section 2 introduces the applications and classification of the existing clustering algorithms and neuron models as prior work. The inhibitive pulse-coupling model and clustering results are detailed in Sect. 3. After describing the basic physical requirements for networks in Sect. 4, the GCC method is listed step by step in Sect. 5. Section 6 focuses on the analysis and simulation of the proposed GCC method and Sect. 7 summarizes the work and discusses some of the future work.

2 Prior work

One of the most classical paradigms of distributed computing is the clustering [6]. Clustering in wireless sensor network has been studied a lot because it is very useful for network scaling, routing, and energy saving. Scaling involves large numbers of nodes being grouped into smaller number of clusters; the head node in each cluster serves as the manager, which makes the management of a large scale sensor network much easier. Routing involves member nodes sending the data to their head nodes; the data packets are transmitted only by head nodes hop by hop to the base station, which can avoid the high energy cost and serious interference caused by directly transmitting. Energy saving involves, when there is no transmission needs, the clusters that can sleep alternatively and save energy. In traditional clustering algorithms, the cluster divisions and formations are led by the cluster head. Nodes which have closer distance to the designated head node are grouped together. The only criterion for deciding which cluster a node belongs to is its distance to the head node or the communication cost. The main work in traditional clustering algorithms is to choosing proper cluster head. Some algorithms choose cluster head based on the residual energy [7], some are based on node degree [8–11] and some are based on the combine weights of several critical factors [12]. All of these algorithms partition the network area geometrically and the correlation between clusters formations and the sensing environment or sensing data are not considered.

The work in [13] architects sensor networks as virtual databases such that each sensor node is a 'living' data tuple source. Therefore, the node clustering problem can be solvable by the data clustering method. Although they already have numerous mature data clustering algorithms, most of them need large amounts of data reciprocation, complex calculation and long iteration times, which are not suitable for the energy and computing ability constrained sensor network. The pulse-coupling oscillator model inspired from the mutual interaction among neurons can be considered a data clustering method. Its calculation is simple, and there need not be complex data

Table 1 Different converging results of pulse-coupling oscillator model

Concave down	Instant coupling	Excit	→ Sync
		Inhi	→ Async
	Delay coupling	Excit	→ Unstable clustered
		Inhi	→ Stable clustered
Concave up	Instant coupling	Excit	
		Inhi	→ Clustered and degrade
	Delay coupling	Excit	
		Inhi	

communication but simple signal and finite iteration rounds. All these factors suggest its suitability to the multi-node computing system.

Recently, the pulse-coupling oscillator models are studied and introduced in the field of sensor networks. However, most of the works focus the applications on time synchronization. Mirollo and Strogatz [14] prove that through very simple reactive adjustments of the node phase after hearing the fire pulse, the phases of all oscillators would converge to a global synchronicity, regardless of the number of nodes and their initial states. Lucarelli and Wang [15] lift the all-to-all communication requirements indicated in [14] and prove that the convergence can be achieved by local coupling. Hong and Scaglione [16] show that when the coupling strength and node density are big enough, there will be some avalanche effects and the whole network locks to synchronicity immediately because of the continuous firing. Our previous work [17] gets rid of the swing actions in the coupling procedure by predicting the final converging direction and improves the synchronizing speed and energy efficiency. References [18] and [19] take into account more realistic factors of the wireless environment. Instead of responding individually and instantaneously, nodes in [18] accumulate the incoming pulses in the past period and do the phase adjustment once in all at the beginning point of the next period, while [19] enlarges the period from $1T$ to $2T$. When the node gets a pulse at some point of the first T, it will react to the pulse at the same point of the second T. The extended T is used to buffer the transmission delay.

All of the above-mentioned works use the same Integrate-and-Fire (IF) oscillating function and coupling form, which is monotonically increasing, concave down, excitatory adjusted and instantly coupling. Actually, in dynamic systems and brain neuron systems, there are many variations on IF and coupling forms. The integrating function curve can be concave down or concave up. When doing coupling, the phase can jump forward (excitatory) or backward (inhibitory). The node coupling dynamics activity can either include delay or not. Table 1 lists the different results along with different choice of these basic elements.

3 Mathematical model

Considering the transmission delay and final stability, we choose the concave down, delay coupling scheme with inhibitory coupling as the core model.

3.1 Delayed inhibitory pulse-coupling model

Although there are many variants of oscillator models for different research purposes [20, 21], here the basic Leaky IF model is used because it is a widely used, simple yet effective model. The 'IF' means the potential x_i accumulates by integrating until up to some threshold x_{th}. Then, the oscillator fires and emits a pulse signal $\delta(t)$. After firing, the potential falls down to zero and starts to integrate again in the next cycle.

The presentation of the following mathematical discussion follows the literature and adds additional clarity about the equation's characteristics.

The integrating speed of the potential is presented by the following differential equation:

$$\frac{dx_i(t)}{dt} = -x_i + I_0 + \sum_{j=1, j\neq i}^{N} J_{ij} P_j(t) \tag{1}$$

The change of potential is decided by the regular integrating speed and the instant leaps caused by the incoming pulses.

The first part of (1),

$$\frac{dx_i(t)}{dt} = -x_i + I_0$$

shows that when the oscillator is isolated, the potential integrating speed is decreased as the potential value increases. Here $I_0 > x_{\mathrm{th}}$ controls the natural period of the oscillator. $\frac{dx_i(t)}{dt} > 0$ and $d(\frac{dx_i(t)}{dt})/dt < 0$ mean the curve of the potential value is monotonically increasing and concave down.

When oscillator i is in a network and can receive fire signals from neighboring oscillators (j), its potential instant speed is adjusted by

$$\sum_{j=1, j\neq i}^{N} J_{ij} P_j(t)$$

Here J_{ij} is the coupling strength between nodes i and j.

$$P_j(t) = \sum_m \delta\bigl(t - \tau_j^{[m]}\bigr)$$

represents all pulse signals from node j in the past, in which $\delta(t)$ is the Dirac delta pulse function and $\tau_j^{[m]}$ is the mth fire time of oscillator j.

Figure 1 shows the curve of an oscillator potential (Y axis) along with its phase (X axis) in an IF model. The potential is written as a function of phase ϕ:

$$x_i(t) = f\bigl(\phi_i(t)\bigr) \tag{2}$$

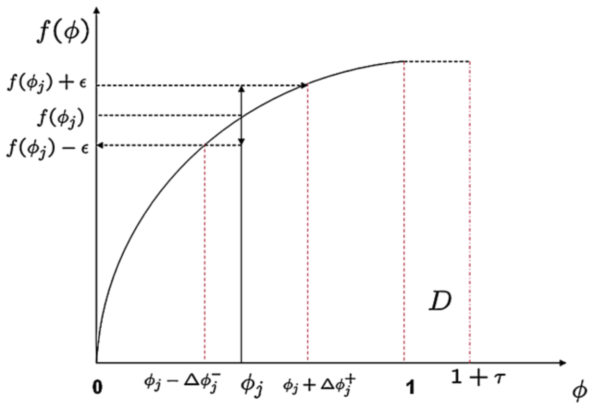

Fig. 1 Potential $f(\phi)$ and phase ϕ adjustment during excitatory/inhibitory coupling

in which phase is defined in the range of $[0,1]$ and the function f meets all the requirements of potential ($f'>0$, $f''<0$). Here, set $x_{\text{th}}=1$ so $f(\phi)\in[0,1]$ and $f(0)=0$, $f(1)=1$. Because of the continuity and monotonicity of f in $[0,1]$, the oscillator potential and its phase are affine one-by-one. The variable phase can reflect the characteristics of the oscillators potential, so in the rest of the paper we use phase ϕ as the main measurement to investigate the dynamics of the oscillators in the networks.

3.2 Clustering in delayed inhibitory coupling

When there is no coupling, the phase moves at the speed

$$\frac{d\phi_i}{dt}=\frac{1}{T} \tag{3}$$

when at time t oscillator i fires, all its neighboring nodes will react to the fire signal at time $t'=t+\tau_0$. Here τ_0 is the delay time between firing and reacting. And their phases will adjust according to

$$\phi_j(t'^{+})=\begin{cases} f^{-1}(f(\phi_j(t'))+\epsilon_t)=B, & \text{if } 0<B<1 \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

in which

$$\epsilon_t=\frac{n_0(t)}{n}\epsilon$$

Because the phases are bounded in $[0,1]$, if B's value is less than 0 or greater than 1, it should be regarded as 0. n is the network node number and $n_0(t)$ is the number of nodes which fire simultaneously at time t. As converging degree increases, n_0 will increase from small value (1 or 2) to its maximum value, which is the number of nodes in each cluster under the totally converged state. ϵ is the coupling strength. Factor $0<\frac{n_0(t)}{n}<1$ controls the amount of phase adjusting and helps to get the clustered result. When $\epsilon>0$, the potential will jump up by ϵ_t amount. When phase jumps

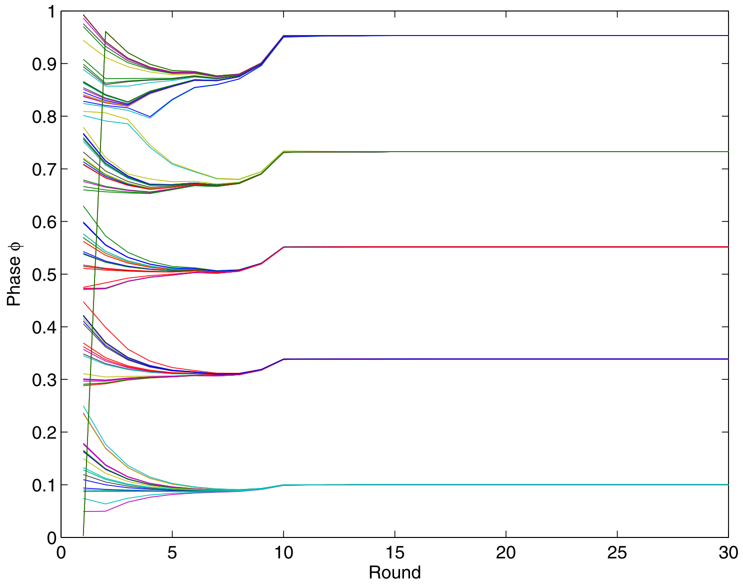

Fig. 2 The converging procedure and clustered result under delayed inhibitive coupling

forward a step, the integrating period will be shortened. The coupling is excitatory. When $\epsilon < 0$, the potential falls down, the phase jumps backwards and the period is elongated. The coupling is inhibitory (Fig. 1).

References [22] and [23] observed that when the coupling is inhibitive and always has τ delay after firing, the oscillators in the whole network will converge to several clusters. All members in one cluster share the same phase. Between clusters, each pair of the clusters have a fixed phase offset. They are phase-desynchronized (Fig. 2). And this sync/desync state is stable because either the outside noises or minor phase deviations cannot break down the clustered state. Besides, empirical data show that the number of converged clusters m is approximately inversely proportional to the twice the length of delay τ:

$$m \approx 1/2\tau \tag{5}$$

Regarding the converging reason why the clusters are of a size 2τ, Ref. [24] gave out a heuristical proof of two nodes' absorbtion and repulsion: whenever two oscillators' phase difference $|\Delta\phi|$ is less than delay τ, after finite iterations, they will for sure end up with synchrony state; when $\tau < |\Delta\phi| < 1 - \tau$, their phase lag will lock onto some fixed distance which equals $\Delta\phi_0 \in [\tau, 1 - \tau]$ and they will never synchronize, which is called the antiphase stable state. When there are more than two oscillators, Ernst et al. [23] explained that the system fixed point's attractive radius is around τ size. So "… the phase axis is partitioned into intervals of size $\approx 2\tau$, each of which being a basin of attraction for one cluster." Although numerous simulations confirmed this phenomenon, until now there is no strict mathematical proof on the convergence when the number of nodes is greater than 2.

The above discussion leads to two significant properties: (1) oscillators are clustered in phase, and (2) synchronization and desynchronization can be achieved at the same time. These are two attractive characteristics for distributed networks.

4 Physical requirements of a network

Before formally introducing the new clustering method, we first point out the physical requirements on system nodes and distributed networks.

(1) Each related node in the computer systems is equipped with hardware or software oscillator with fixed frequency. All oscillators should have the same oscillating frequency but different initial phases are allowed.

(2) There is no limitation on transmission media. The coupling signal can communicate in either wired or wireless networks. However, due to existing mature time synchronization protocols in wired networks together with the current emphasis on wireless networks, the remainder of this section is devoted to wireless technology.

(3) The firing signal used for causing coupling among nodes should be short, fast and low-cost. The only purpose of the signal is to inform other nodes about their firing time point, so the signals need not to contain any source or target information. At the first stage of converging, the signal fires chaotically and frequently, so the firing signal should be fast and low-cost for the energy and channel interference considerations.

Several options are available. One option is based on Ultra Wide-Band (UWB) pulse radio [25, 26], which can be configured in a low bit rate long-range transmission configuration [27, 28] to meet the requirements of the firing signal. A second option is based on Free Space Optical (FSO) pulse communication system, which supports long-distance point-to-point node communication and moreover can be configured to support broadcast [29]. In addition, since Radio Frequency (RF) systems are common, a more practical option is to use the short message supported by the existing transmission protocol, such as a 56-bit short-term synchronization head file in IEEE 802.11g [30]. The detection of a pulse message is dependent on which technique is deployed: for example, in optical pulse systems the detector (e.g. photodiode) can be used to detect multiple pulses [31, 32].

(4) All nodes locate in a single cell. To guarantee the convergence of the oscillators to the clusters, the system requires that the coupling is of the all-to-all form, which means that when one node fires and emits a signal, all nodes in the network should receive it and react to it. Simple broadcast in single cell is sufficient for all-to-all coupling. When the network scale is beyond one hop, relay broadcasting of the firing signal is required at the peripheral of the cell. This relay will increase the transmission delay. The more relay times, the lager indetermination there will be on delay value, which is caused by the message lost or interference, etc. A key requirement for stable clustering result is that all nodes should have the same length of delay. The increase of indeterminate delay will make the τ's setting and adjusting more complex. As this is beyond the scope of this paper, only the single cell case will be discussed.

Although in this paper some examples and simulations are given in the wireless sensor network environment, the GCC method is not only limited to it. All the multi-node computing systems which meet these requirements are applicable by this method.

5 General Criteria-based Clustering (GCC) method

In properly equipped networks, the GCC method can self-organize the system nodes into expected number of clusters. This section gives the detailed steps of the method.

Step 1. Clustering criteria selection and mapping
The significance of the GCC method is that its clustering criteria is general and application dependent. This means that any node-related data or properties may be selected as the clustering criteria (e.g., in WSN, the node's geography information, identification number or connecting degree, etc.). The selection is dependent on the purpose for the cluster formation, which is often associated with the application. Multiple criteria may be chosen for different reasons thereby enabling different possible cluster re-formation. The GCC builds a connection between application requirements and clustering construction due to the criteria selection. Such a connection means that a known objective may lead to a criterion selection, or a criterion can be firstly selected that results in some benefit. For example in WSN, for the objective of network work load balancing, the node's residual energy amount can be worked as the clustering criterion. Nodes with similar residual energy level are thus grouped, and according to the different energy levels, the cluster members can adjust the sampling and reporting rate so as to balance the load and extend the network lifetime. Or conversely, if using node sensed content as the clustering criteria, then it may be easier to perform data aggregation inside clusters whose members have similar content data. The final condensed information would be shorter and cost less transmission energy.

There are some constraints on the selection. The criteria properties should be numerable and the value be bounded in some range, because the final selected criterion is mapped as the node initial phase which is a one-dimensional scalar variable in the domain of [0, 1]. However, in the case when the selected criterion consists of non-scalar data (like X,Y-coordinates), a data dimension transformation is required. Recently there is some mathematical work on multi-dimension phase convergence [33]; however, at this time it may not be suitable for practical system uses.

All the qualified node data or properties form a criteria candidate set $C = \{C_1, C_2, C_3, \ldots\}$. From the set, one or several candidates may be appropriately chosen by a user or automatically in an automatic application. When mapping the selected criterion as the initial phase ϕ, the network should be "aware" of the possible minimum and maximum values of the criterion as boundaries. The values in [min, max] are normalized into [0, 1] (see row 1 in Table 2). These boundary values are calculated prior to the deployment or obtained from the broadcast message before clustering starts.

Step 2. Parameter setting
After choosing the appropriate clustering criteria, the next step is to set parameters in both the physical and mathematical systems.

After obtaining the initial phases, in order to calculate the consequent phase values at some specific time, the moving speed of the phase is needed. As described in Sect. 3.2, the ordinary moving speed of the phase without coupling is decided by the oscillating period T (see (3)). The oscillating period T takes the same value T_0 in both the physical system and the mathematical model. The value of T_0 can be set rather freely under the constraint of the system delay and the expected cluster number m, which is discussed in the following paragraphs. Similarly, the length of T_0 will influence the whole system converging time and the corresponding energy consumption. The latter Step 4 will discuss this influence.

Delays are inevitable in most real networks, especially in wireless networks. The delayed inhibitory coupling model considers the transmission delay directly, in which after some nodes fires, instead of reacting instantaneously, receivers wait for some time and then react. In real networks, the transmission delay mainly includes: transmitting delay t_{tx} in the sender, propagation delay t_{pg} in the air and decoding delay t_{dec} in the receiver [19]. Compared to the t_{tx} and t_{dec}, the propagation time is small enough to neglect. On the other hand, t_{dec} is measured by the receiver itself and thus is known. The only unknown item is t_{tx}. If the exact value of t_{tx} for every signal is needed, then the exact timestamps can piggyback on the signal message like in Ref. [34]. However, in the GCC method, all the fire messages are the same short, pulse-like signals. Their emitting and receiving can be done by hardware or firmware using almost the same time. If the precision requirement is not very high, for simplicity, the transmission delay of all nodes are considered the same. $t_0 = t_{\text{tx}} + t_{\text{dec}}$ represents the total delay for a firing signal between its emission and reception. t_0 is the minimum allowed delay length and the real delay value t_τ should not be shorter than t_0: $t_\tau \geq t_0$. Normalize the time delay for system model:

$$\tau = \frac{t_\tau}{T_0} \tag{6}$$

Considering $\tau \in [0, 0.5]$ and the relation in (5), the delay value also sets the floor boundary of the real oscillating period T_0.

$$T_0 \approx 2mt_\tau \geq 2mt_0 \tag{7}$$

The coupling strength coefficient ϵ measures the inhibitory phase adjustment amount when receiving a firing signal. The desynchronized grouping phenomena and the relationship of (5) are only tenable below some coupling strength threshold. The coupling strength is also critical for the converging speed and final cluster number veracity. We will discuss the empirical choice of the value of coupling strength in the simulation Sect. 6.2.

Table 2 shows the initial phase mapping and the relationship between the physical parameters and the system parameters.

Step 3. Logical clustering

After regarding every computing node as an oscillator and mapping the selected criterion property as the oscillator's initial phase value, based on the theoretic results and analysis in Sect. 3, if the oscillators' potential and phase variables follow the preset oscillating function and execute the inhibitive coupling with the same delay

Table 2 Mapping from Physical to System parameter

Physical parameter	System parameter
Criteria property $[min, max]$	Phase ϕ [0, 1]
Oscillating period T_0	T_0, $\frac{d\phi}{dt} = \frac{1}{T_0}$
$t_{delay} = t_{tx} + t_{pg} + t_{dec} \approx t_{tx} + t_{dec} = t_0$, $\quad t_\tau \geq t_0$	$\tau = \frac{t_\tau}{T_0} \in [0, 0.5]$
Cluster number m	$m \approx \frac{1}{2\tau}$
	Coupling strength ϵ

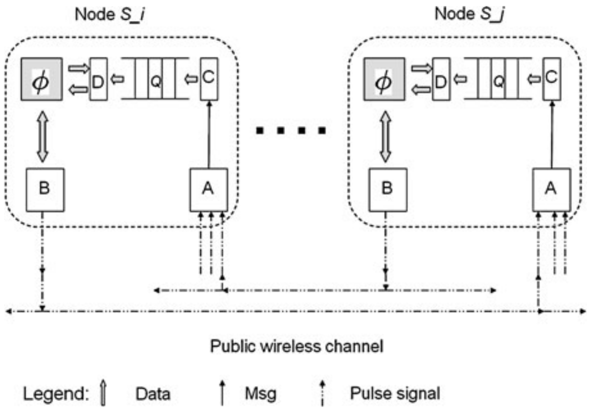

Fig. 3 Two level concurrency among nodes and inside each node

length, after finite iterations, the nodes will be grouped into several clusters by their self-converged phase values (Fig. 2). The following presents the clustering algorithm operated on each computing node.

During the whole clustering procedure, each node mainly executes two operations: periodical self-oscillating and coupling with other nodes. The couplings among the nodes are mutually triggered by the same anonymous pulse-like signals, so each node also operates two assistant signal-receiving and broadcasting processes. Whether the self-oscillating or coupling, the operation results are embodied on the node's phase value ϕ. There are four processes concurrently executed on each computing node: A, channel listening; B, periodical phase value self-increasing and resetting; C, signal receiving; and D, coupling-caused phase value adjustment. Processes A and C are asynchronous communication processes; the B and D perform concurrent read/write computations on the shared node phase ϕ, that is, the ϕ is stored in a shared memory which is accessible by these processes. In addition, due to the delayed coupling model, the node uses a queue structure Q to store the waiting signals between processes C and D (Fig. 3). The node's local clock time is used by all the processes. Figure 3 illustrates the four processes working function inside each node, and Algorithm 1 gives specific details about the required calculations.

Algorithm 1 Logical Clustering()

```
Require: Each node have the same $f$, $\epsilon$, $\tau$, $T_0$, $t_0$ value and different initial phase
    value $\phi_i$
Ensure: System stable clustered
 1. repeat
 2.     (Process A) Listen to the public wireless channel
 3.     (Process B) Increase the phase value $\phi$ at the speed of: $\frac{d\phi}{dt} = \frac{1}{T_0}$
 4.     if ($\phi = 1$) then
 5.         Broadcast Firing_signal()
 6.         Reset $\phi = 0$
 7.     end if
 8.     (Process C)
 9.     if (Heard firing signals) then
10.         Record current local clock time $t_k$ and calculate the expected
            reacting time $t_{rk} = t_k + (\tau \times T_0 - t_0)$
11.         Estimate the firing number $n_0(t_k)$
12.         Add $t_{rk}$ and $n_0(t_k)$ into the waiting Queue
13.     end if
14.     (Process D)
15.     if (The Queue is not empty) then
16.         Read the first item in the Queue
17.         Obtain the upcoming reacting time $t_{rk'}$ and associated $n_0(t_{k'})$
18.         if ($t = t_{rk'}$) then
19.             Read the current phase value $\phi(t)$
20.             Adjust the phase value using (4) with $\phi(t)$ and
                $n_0(t_{k'})$
21.         end if
22.     end if
23. until Detect fixed number and strength of fire signals sequence in continuous
    periods
```

When not performing the coupling, process B in real-time updates ϕ at the speed of $\frac{d\phi}{dt} = \frac{1}{T_0}$, which is an instance of (3). When process B observes the phase value arrives at the threshold value of $\phi \geq 1$, it broadcasts a firing signal to the whole network, meanwhile updates phase as zero $\phi = 0$. Process A continues listening to the public wireless channel and detects if there are any firing signals, and its detected results are only reported to process C. When the node detects firing signals, process C is activated. It records the current node local clock time t_k and estimates the firing numbers of the signal $n_0(t_k)$. The expected reacting time $t_{rk} = t_k + (\tau \times T_0 - t_0)$ and the firing number $n_0(t_k)$ are added to the end of the reacting queue Q of this node. After that, process C is suspended until the next firing signal is reported. Once the queue is not empty, process D reads the first item in Q and obtains the next expected reacting time $t_{rk'}$ and associated $n_0(t_{k'})$. Process D waits until the local clock time $t = t_{rk'}$, then it reads the current phase value $\phi(t)$; using (4) and $n_0(t_{k'})$, it calculates the new phase value $\phi(t^+)$ after the coupling and updates this new value into the shared memory ϕ storage. Termination of all processes occurs when a sequence of firing signals with fixed number and fixed strength are repeatedly reported to process D (via the queue) for continuous several clock periods.

Both processes B and D can access the ϕ memory exclusively, which means that when process B is increasing or resetting the ϕ, process D cannot do the adjustment on ϕ, and vice versa. The read&write in both processes (Algorithm 1, lines 3, 6 for process B; lines 19–20 for process D) should be transaction operations. Because the speed of phase self-increasing is quick (simple adding or resetting), process B's memory holding does not influence process D much. However, process D's processing time is relatively longer than B's; here we set that the phase self-increase updating interval of B is always longer than D's accessing time for ϕ. Giving this criterion valid, the whole concurrency works. Due to the fact that phase self-increasing and resetting are regular actions for the node, under the same circumstance, process D always has higher accessing priority than process B: specifically, *Coupling_adjust* > *Normal_shift* > *Reset*.

From Algorithm 1 and Fig. 3 we can see there is a high-level concurrency among the network computing nodes too. All the nodes operate individually and there is no central control message or data flow. The operations on the computing nodes are parallel and their phase variables' interactions are mutually triggered by each node's firing signal. The algorithm is distributed and the whole system is totally autonomic.

The clusters are divided based on the initial phase which is also the criteria properties distribution. The produced cluster members share the similar criterion property values, but their physical location may or may not be near as those in common traditional clustering approaches. The cluster members are logically related and so the formed clusters are called Logical Clusters (LCs). Because the clustering is totally self-organized, there are no cluster head nodes either before or during the clustering procedure. The logical clusters' physical shapes are also more diverse than the normal ones from the existing algorithms.

Step 4. Period elongation and reverse mapping
After the system converges, each cluster acts as a whole and m clusters fire alternately in one period. The firing pulse number decreases substantively. The superposed pulses from simultaneous firing cluster members are used to maintain the clustering topology of the network and keep it stable. Because of the independence of the clustering method, the cluster head election or rotation inside will not influence the macro network cluster distribution. Without special requirements, the network re-clustering interval can be very long.

During the procedures of converging, a small period is preferred. With the smaller period, the coupling happens more frequently and the system converges more quickly, but after the converging becomes stable, if the system still keeps the short period, the emitting of maintenance pulse will be too frequent. Besides, the computing nodes also need sufficient time to fulfill their other duties between fires. Therefore, the ideal situation is to use small T_0 during the converging procedures, increase the converging speed and decrease the time cost, and use an appropriate longer T_0 after converging becomes stable, firing occasionally to maintain the clusters, decreasing the maintenance cost. This needs to elongate the period T_0 after converging smoothly without disturbing the stable network state. To obtain this, there are two points to note. First, the new period should be integer multiples of the old period, like $2T_0$, $5T_0, \ldots,$ etc.

Second, the system precision should be high enough so as not to split the already converged clusters. This period enlarged function will be tested in the simulation section (Sect. 6.3).

As mentioned in the Introduction part, the members are phase-synchronized inside a cluster. If the application requires time synchronization among nodes, the phases can be reversely mapped to some accumulated time variables. Because the member nodes' phase and frequency are all the same, the transformed timescale must be the same as well, so all these computing nodes are time-synchronized.

6 Simulation results and discussion

The simulation is divided into three parts. First we check the feasibility of GCC method and demonstrate the produced novel logical clusters. Then we discuss the cost of GCC method. Finally, the stability of the clustering results under enlarged period is verified.

The basic simulation environment is set to meet the requirement in Sect. 4. Three hundred nodes are randomly deployed in a 100×100 meter size field. Each node uses the uniform oscillating function $f(\phi) = \frac{1}{b}\ln[(e^b - 1)\phi + 1]$ [14], in which $b = 3$ measures the extent to which f is concave down. Here we assume the transmission media is wireless. And the firing signal is chosen to be the 802.11g short preamble—a specific sequence of pulses which has the length of 56-bits. The field is set as a single cell where firing signals are directly broadcasted and received and no relay work used. If needed, there will be other environmental parameters given in each part.

6.1 General-criteria clustering

To check the generality of the GCC criteria, here we choose three kinds of representative criteria as examples.

(1) The location data of each node.
In Fig. 4, the geography data criteria is the angle to the base line of a polar coordinate (defined by point $(20, 40)$ and line $y = 40$). The produced polygon clusters are similar to the normal ones except that there is no predesignated head nodes. This example shows that the GCC method is compatible with the traditional clustering algorithms. Figures 5 and 6 select the distance to the field point $(50, 50)$ and the outside field line $y = 1/2x + 200$, respectively, as the criteria and map the distance values as their initial phases. The formed cluster shapes are circles and strips respectively. As discussed in Step 1, the physical borderline of each cluster is clear and these kinds of clusters are suitable for hotspot surveillance applications. Inside the cluster, head centric star topology may not be suitable any more. Instead a chain topology like in Power-Efficient Gathering in Sensor Information Systems (PEGASIS) [35] is more practical for strip shaped clusters. The duty of the head node can shift along the chain for each round.

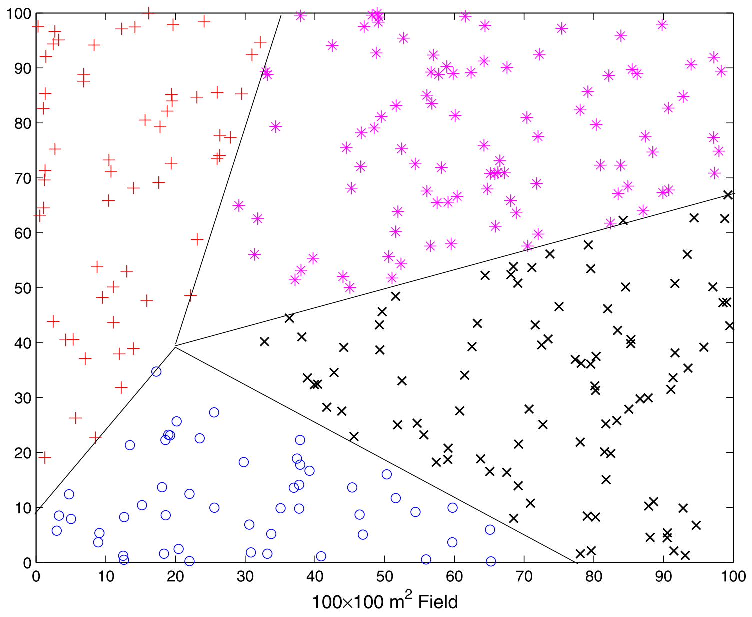

Fig. 4 Normal polygon-shaped clusters based on angle to (20, 40) and $y = 40$

(2) Content-based clustering criteria.
In a $702.5 \times 310 \times 470$ cm^3 size room, 80 observing points, which form a $4 \times 4 \times 5$ three-dimensional grid of sensors, are set to measure the indoor air temperature of different places. At the right corner of room ((690, 5, 275), star mark in Fig. 7), there stands an air conditioner blowing out cold air. The nearer to the air conditioner, the lower the air temperature is. The temperature values are determined from a simulation using a simple linear distribution. The temperature values are mapped to the initial phase and system parameters ($\epsilon = 0.3$, $\tau = 0.1$) are set as discussed in Sect. 5 steps. The 80 nodes are converged into 4 clusters. Figure 7 shows the three-dimensional clusters based on the node sensed content, in which the contour surfaces are shown to highlight the clusters.

(3) The unique node ID.
This produces scatter clusters as shown in Fig. 8. The scatter cluster shows the key idea of logical cluster. The clusters are overlapped in a physical field and cluster members are scattered everywhere. Intuitionally, these clusters have no physical meanings, but they are logical related and share some common points among cluster members. This concept is similar with an online social network. The virtual social group share the same interests and based on that new ideas may come up.

From Figs. 6 and 7 we notice that some of the highest phase nodes and lowest phase nodes are clustered into one group. This is because the node phase is a modular variable in the oscillator model: the phase value near 1 also means near 0. Given the phases of nodes whose criterion property values near minimum or maximum are close, there is the possibility to be clustered into one group. This will not be a problem

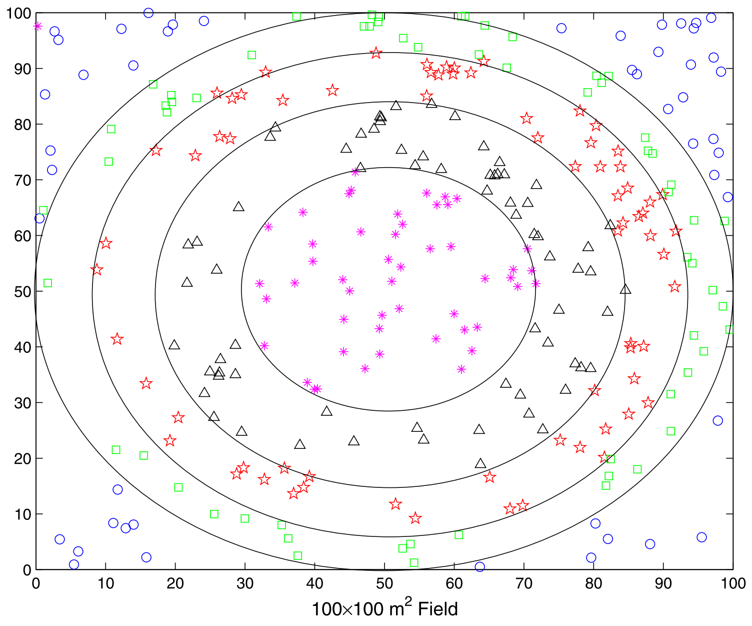

Fig. 5 Concentric clusters based on distance to point (50, 50)

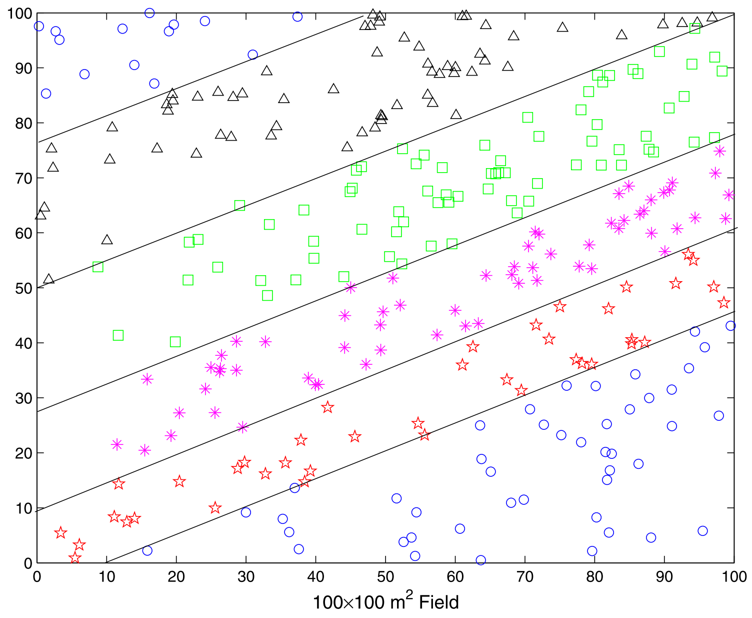

Fig. 6 Strip clusters based on distance to line $y = 1/2x + 200$

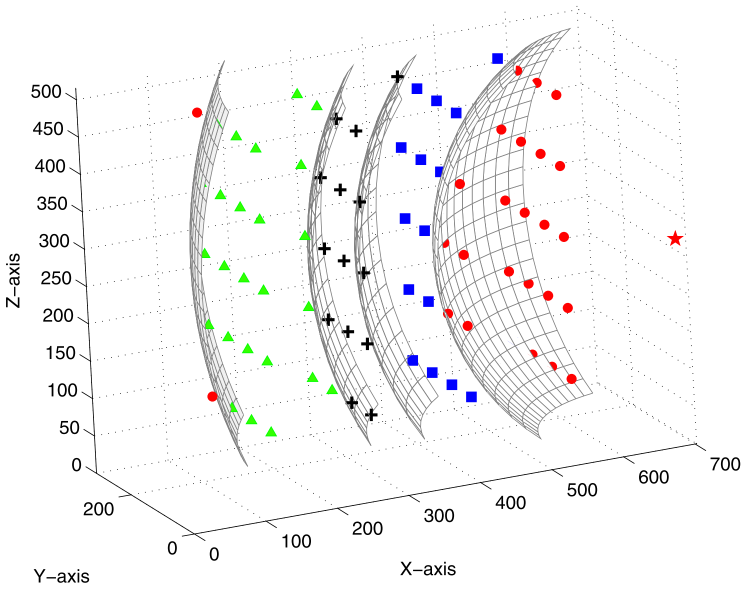

Fig. 7 Three-dimensional clusters based on indoor temperature

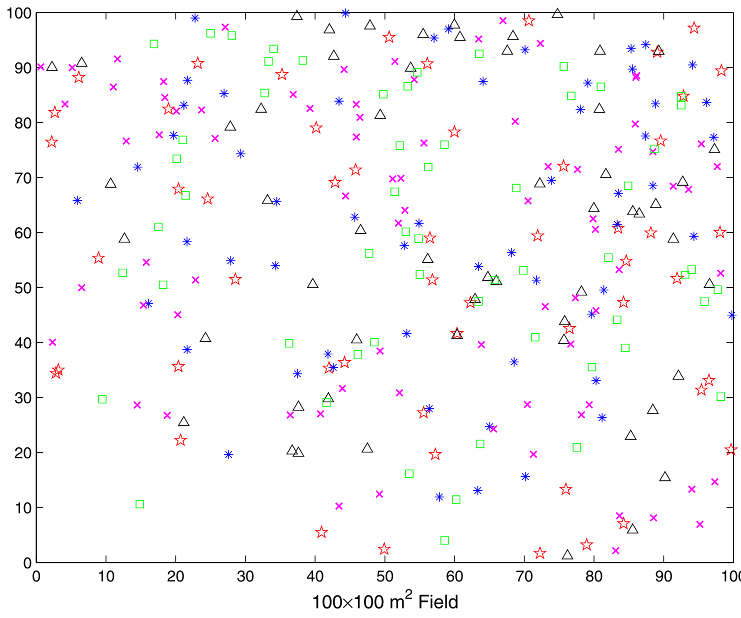

Fig. 8 Scattered logical clusters based on node ID

when the logical cluster is also modular (like the example of life-surviving temperature clustering, both the highest and lowest temperatures are not suitable for life to survive, clustering them into one group is reasonable). However, in other situations, these two contradictory classes may cause confusion. They will need to be subdivided inside the cluster. In some cases, with the help of physical base parameters, it would be easy to differentiate the two class nodes (like in Fig. 6, the physical location is substantial different between the two class nodes). A more general way is based on each node's firing times. Based on the property of the oscillating function f and the phase coupling formula (4), the firing order of nodes never changes during the system converging procedure. So although the two classes of nodes may converge and they fire together, the higher phase nodes always fire one more time than the lower nodes due to their different initial phase values. By comparing their firing times, the nodes can subdivide themselves.

6.2 Parameter and cost analysis

The previous discussion was focused on the possible cluster results. Now the clustering costs are given in greater detail. Here cost refers to the time and energy consumed during the procedures from random initial distributed network status to the completely clustered state.

The time for all nodes in a network fire once are defined as one convergence iteration (I), and it is an important merit to measure the algorithm's time and computing complexity. The length of one iteration I is a little longer than one oscillating period T because of the backward jump of phases in the inhibitive coupling. For the all-to-all coupling scheme, in one round every node fires once and reacts to the firing signal at most n times. It is because the signals that emit simultaneously are only received and reacted once. Assuming the system converged in I_0 iterations, then it costs around $I_0 \times T_0$ time to accomplish the clustering, and during that every node on average fires I_0 times and reacts less than $I_0 \times n$ times.

There are several factors that may influence the converging speed: the initial phase distribution, the coupling strength ϵ, the extent to f's concavity b and the node number n. The initial phase distribution is determined by the clustering criteria. f's concavity b has the same effect with coupling strength ϵ, both of which influence the phase jump amplitude in the coupling. Figure 9 shows the relationship between converging iteration number I and coupling strength ϵ under the different node numbers n. In Fig. 9, the four lines take the randomly uniform initial phase distribution and the resulted data on them are averaged over 500 times initial phase realizations. As predicted, the stronger the coupling strength is, the faster the system converges. The node number has little influence on converging speed, because the lines with different node numbers almost superpose. Therefore, we suppose the converging iteration number I is independent to the system node number n, and the computing complexity of GCC is $O(n)$.

As is shown in Fig. 9, when the coupling strength is strong, it costs less rounds and the associated energy for each node to achieve convergence. However, too strong on adjustment in each step may combine the neighboring clusters which should have been separated. This brings the control precision lapses of the expected cluster number. Therefore, the value of ϵ is critical to converging speed, energy consumption and the cluster number veracity. Next we will show how to decide a proper coupling

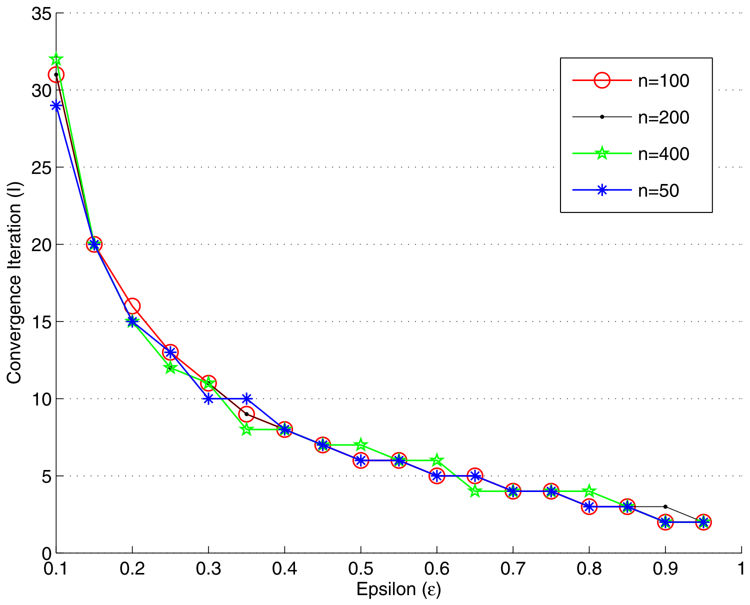

Fig. 9 Converging round with respect to coupling strength under different node number

strength ϵ in a network to meet the high veracity of the clustering number with possible low energy consumption at the same time. To calculate the energy consumption, the first order radio model [5, 36] is used. Each senor node will consume the following E_{Tx} amount of energy to transmit an l-bits message over distance d:

$$\begin{aligned} E_{Tx}(l,d) &= E_{Tx\text{-elec}}(l) + E_{Tx\text{-amp}}(l,d) \\ &= \begin{cases} lE_{\text{elec}} + l\epsilon_{fs}d^2, & d < d_0 \\ lE_{\text{elec}} + l\epsilon_{mp}d^4, & d > d_0 \end{cases} \end{aligned}$$

E_{Rx} is an amount of energy to receive this message:

$$E_{Rx}(l) = E_{Rx\text{-elec}}(l) = lE_{\text{elec}}$$

In the model, the radio dissipates E_{elec} to run the transmitter or receiver circuitry and E_{amp} for the transmitting amplifier. If the distance is less than a threshold d_0, the free space model (ϵ_{fs} and d^2 power loss) is used; otherwise the multipath (ϵ_{mp} and d^4 power loss) is used. The values of these variables are labeled in Table 3, and here $l = 56$ bits for GCC.

Here we use two ratio values to measure the node consumption situation and cluster number veracity. *ECT* denotes the energy consumption ratio which is the rate that average energy cost for each node to its maximum energy ($2J$). *ERT* denotes the cluster number error rate which is defined as $ERT = |m' - m|/m$. m is the expected final cluster number according to (5) and m' is the real cluster number obtained from

Table 3 Simulation environment

Type	Parameter	Value
Network	Network grid	From (0, 0) to (100, 100)
	Initial energy	2 J/battery
Radio model	E_{elec}	50 nJ/bit
	ϵ_{fs}	10 pJ/bit/m^2
	ϵ_{mp}	0.0013 pJ/bit/m^4
	Threshold distance (d_0)	$\sqrt{\frac{\epsilon_{fs}}{\epsilon_{mp}}} \approx 87.7$ m

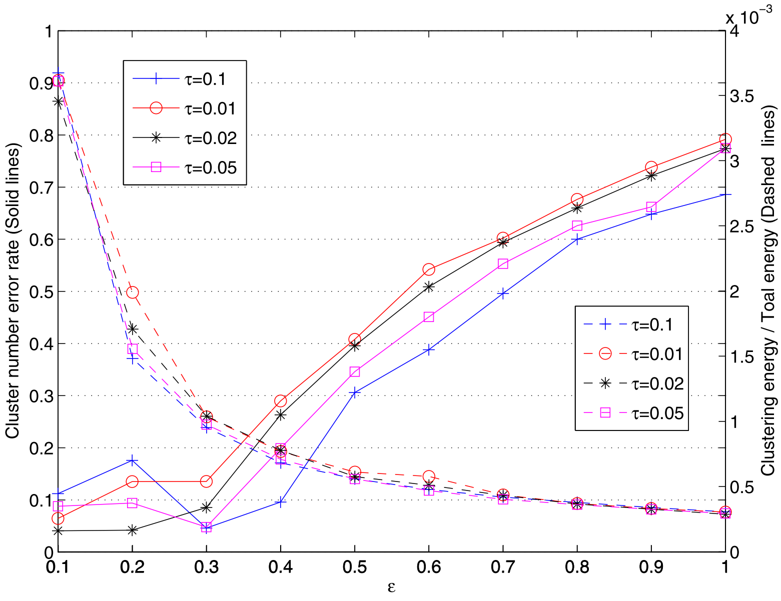

Fig. 10 GCC energy consumption and cluster number error rate with respect to coupling strength ϵ

the simulation. Figure 10 shows the *ECT* and *ERT* of 300 nodes with different delay values τ under different coupling strength ϵ. All the resulting data appearing in this figure and Fig. 11 are averaged over 500 network topologies and initial phase realizations.

In Fig. 10, dashed lines show that the energy cost ratio decreases with the increase of the coupling strength. When $\epsilon > 0.3$, the energy consumption ratio keeps under 0.1% (right axis), and the difference in cluster numbers have little influence on the energy cost. Solid lines show that the cluster number error rate increases with the increase of ϵ. Roughly, when $\epsilon < 0.4$, the error rate can be kept under 30% (left axis). When the cluster number is big ($\tau = 0.01, 0.02$), choosing the coupling strength as small as possible is preferred for obtaining expected m. While the cluster number is not big ($\tau = 0.05, 0.1$), choosing some proper strength ($\epsilon = 0.3$) can obtain the most exact m. According to the empirical data and results in Fig. 10, energy cost ratio for

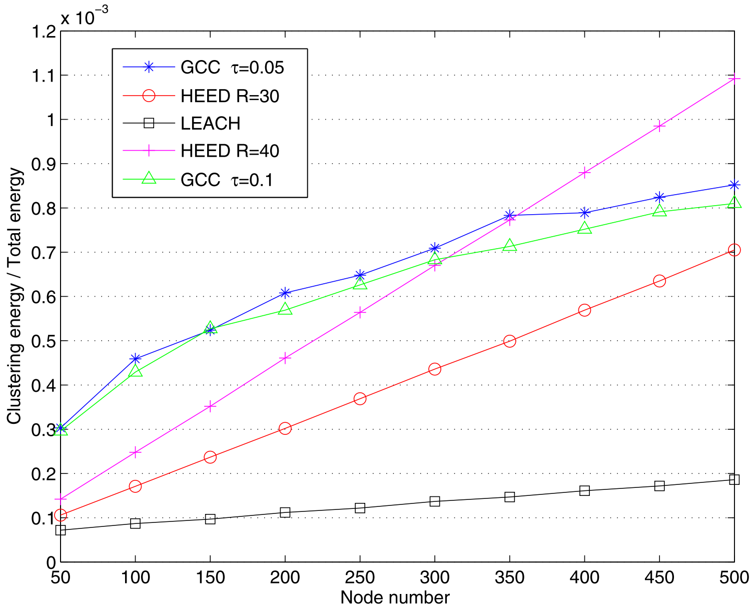

Fig. 11 Energy cost ratio comparison under different node density. (GCC $\epsilon = 0.4$; LEACH 5% nodes as head)

clustering below 0.1% is proper. So ϵ chosen inside the range $[0.3, 0.4]$ is suitable for both energy cost and cluster number veracity in this environment.

Next we compare the energy consumption situation of the GCC method and the traditional clustering algorithms. Two representative distributive clustering algorithms HEED (Hybrid, Energy-Efficient, Distributed Clustering) approach [7] and LEACH (Low Energy Adaptive Clustering Hierarchy) protocol in sensor networks are selected for comparison purposes. For HEED and LEACH, the broadcast packet used in the clustering procedures usually takes the size of 25 bytes = 200 bits. Figure 11 shows the comparison of *ECT* of the three clustering algorithms from sparse to dense network environment. For HEED, the fixed power level is used for intra-cluster communication and choosing min-degree as the second clustering parameter [7]. In GCC, the coupling strength uses the empirical $\epsilon = 0.4$.

For GCC, the energy cost increases with the node density and the different delay values almost have no influence. While for HEED, energy cost linearly increases with the node density, and bigger transmission radius r costs more energy. This is because HEED chooses the best suitable node (highest residual energy, minimum cost) as head node from its neighbors. So when the density increases or neighborhood enlarges, it has to communicate and check more neighboring nodes to decide the head node and this costs more energy. Comparing HEED with GCC, when transmission radius $r = 30$ and $\tau = 0.05$, HEED and GCC both produce around 10 clusters. Every node in GCC costs more energy than in HEED. At $r = 40$ and $\tau = 0.1$, they both produce around 5 clusters. In sparse network, GCC still costs more than HEED, but as the node density increases, GCC's *ECT* gets closer to HEED and becomes lower

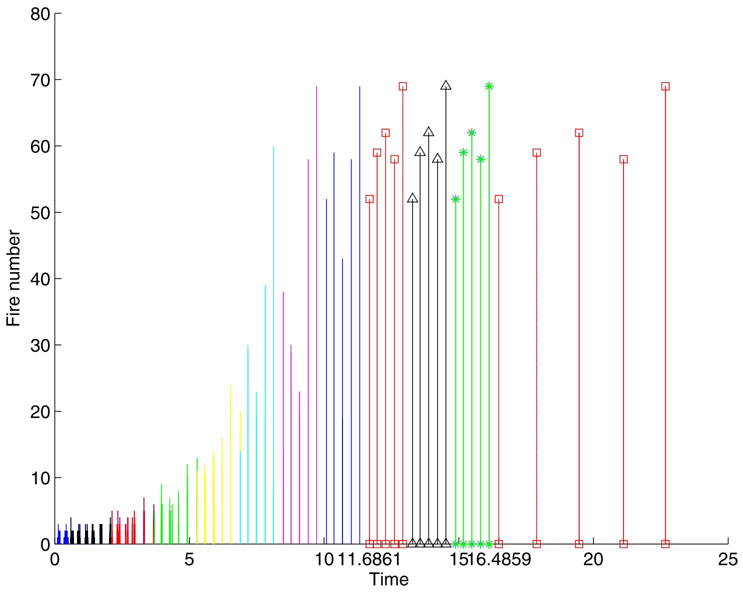

Fig. 12 Fire strength along converging time and extended period after stable. ($\epsilon = 0.35$, $\tau = 0.1$)

than HEED when $n > 300$. This is because at the latter stage of converging, the times of receiving action decrease because of more and more converged nodes. The GCC's computing complexity is slower than linearly increasing. Therefore, the trend is when the network becomes more dense, the GCC's energy performance becomes better. Here we use the most simple version of LEACH, in which the cluster head nodes are chosen based only on the rotation probability. Neither the residual energy nor other requirements are taken into consideration. It costs the least energy among the three algorithms and increases mildly with the increase of the node density. In reality, it will cost more than that but from the energy perspective, LEACH is still superior to HEED and GCC because of its simplicity.

GCC has an acceptable energy cost for clustering, and it can be improved further by the period adjustment stated in Sect. 5 Step 4.

6.3 Stability for period elongation

In Sect. 5 Step 4, we discussed the period elongation. Here we will consider its feasibility and its stability.

Figure 12 shows the system converging procedures and the stability of clustering results when the oscillating period is enlarged. The Y axis records the numbers of superposed firing signals and the X axis represents time. The higher a line means that the more signals are emitted at this time point. The same gray degree (color) lines in the figure represent the same round fires. From the figure we can see that, at first, nodes fire almost all the time in one round. Gradually, there become less fire spots but meanwhile the fire numbers at these spots become bigger. Until the 9th round ($t = 11.6861$), the whole system converges into 5 clusters which fire alternately with approximately the same strength and intervals in one period. Once the system is converged, it will keep the stable clustering status periodically. After three periods ($t = 16.4859$), we extend the system period to 5 times the length ($5T$) by slowing the oscillator phase forward speed by 5 times (the original 5 ticks can be regarded as one). Each cluster keeps the same members and they are still phase-synchronized while their fire intervals are 5 times enlarged. This testifies the stability of GCC and shows that the period elongation is a feasible way to accommodate the different requirements between constructing and utilizing stages.

7 Conclusion and future work

By treating the physical computing nodes as homogeneous oscillators with fixed period, our proposed General Criteria-based Clustering (GCC) method can cluster the computing nodes in both physical and logical ways according to their initial phase values. The oscillator's initial phase value is mapped from some property value of the node, and the selected property data works as the criteria of the clustering. In the GCC method, the criteria can come from any numeric node-related data or properties and its selection also can be influenced by the application level requirements. The method can generate similar physical clusters as those obtained from the existing traditional clustering algorithms and also generate logical related clusters according to selected application requirements. This broad choice of clustering criteria indicates the generality of the method that results in new cluster formations not otherwise existing. We believe that new perceptions about clustering are enabled and hence the GCC method demonstrates broadly applicable potential in future multi-node computing system.

During the course of this research, observations about the performance of GCC are made, some of which have been pointed out in this paper. These include in part the system converging speed, the fine tuning of the control parameters and alternative pulse signal techniques. The further investigation on these items forms the future work.

Acknowledgements This research was supported by Future-based Technology Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2010-0020725). This research also was partially supported by Kyung Hee University under the Young Researcher's Program (20090704).

References

1. Degesys J, Rose I, Patel A, Nagpal R (2007) Desync: self-organizing desynchronization and TDMA on wireless networks. In: Proc of the sixth international symposium on information processing in sensor networks (IPSN2007), Cambridge, MA, USA, Apr, pp 10–20
2. Sekiyama K, Kubo Y, Fukunaga S, Date M (2005) Self-organizing communication timing control for sensor network. Complex Int 12 [Online]. Available: http://www.complexity.org.au/vol12/msid58/
3. Sekiyama K, Suzuki K, Fukunaga S, Date M (2005) Autonomous synchronization scheme access control for sensor network. In: Proceedings of 9th international conference on knowledge-based intelligent information and engineering systems, Melbourne, Australia, Sept, KES 2005, vol 4, pp 487–495

4. Tate J, Bate I (2009) An improved lightweight synchronisation primitive for sensornets. In: Proc 6th IEEE international conference on mobile ad-hoc and sensor systems, Macau, Oct, 2009. IEEE Computer Society, Los Alamitos, pp 448–457
5. Heinzelman W, Chandrakasan A, Balakrishnan H (2000) Energy-efficient communication protocol for wireless sensor networks. In: Proceedings of the 33th Hawaii international conference on system sciences, Havaii, USA, Jan, p 10
6. Myoupo JF, Cheikhna AO, Sow I (2010) A randomized clustering of anonymous wireless ad hoc networks with an application to the initialization problem. J Supercomput 52(2):135–148
7. Younis O, Fahmy S (2004) Heed: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. IEEE Trans Mob Comput 3(4):366–379
8. Kuhn F, Moscibroda T, Wattenhofer R (2004) Initializing newly deployed ad hoc and sensor networks. In: Proceedings of the 10th annual international conference on mobile computing and networking (MOBICOM), Philadelphia, PA, USA, Sept. Springer, Berlin, pp 260–274
9. Amis A, Prakash R, Vuong T, Huynh D (2000) Max-min d-cluster formation in wireless ad hoc networks. In: Proceedings nineteenth annual joint conference of the IEEE computer and communications societies INFORCOM 2000, Tel-Aviv, Israel, Mar, vol 2, pp 32–41
10. Chan H, Perrig A (2004) Ace: An emergent algorithm for highly uniform cluster formation. In: Proc first European workshop sensor networks (EWSN), Jan, vol 2920. Springer, Berlin, pp 154–171
11. Liu Y, Kwok YK, Wang JZ (2008) On scheduling and clustering in hierarchical TH-PPM UWB wireless ad hoc networks. J Supercomput 46(1):58–83
12. Chatterjee M, Das SK, Turgut D (2002) WCA: a weighted clustering algorithm for mobile ad hoc networks. Clust Comput 5(2):193–204
13. Govindan R, Hellerstein JM, Hong W, Madden S, Franklin M, Shenker S (2002) The sensor network as a database. USC Computer Science Department, Tech. Rep. Technical Report 02-771
14. Mirollo R, Strogatz S (1990) Synchronization of pulse-coupled biological oscillators. SIAM J Appl Math 50(6):1645–1662
15. Lucarelli D, Wang I (2004) Decentralized synchronization protocols with nearest neighbor communication. In: Proc 2nd ACM conference on embedded networked sensor systems (SenSys'04), Baltimore, Maryland, USA, Nov, pp 62–68
16. Hong YW, Scaglione A (2005) A scalable synchronization protocol for large scale sensor networks and its applications. IEEE J Sel Areas Commun 23(5):1085–1099
17. Niu Y, d'Auriol BJ, Wu XL, Wang J, Cho JS, Lee SY (2008) Selective pulse coupling synchronicity for sensor network. In: Second international conference on sensor technologies and applications 2008 (SENSORCOMM '08), Cap Esterel, France, Aug, pp 123–128
18. Werner Allen G, Tewari G, Patel A, Welsh M, Nagpal R (2005) Firefly inspired sensor network synchronicity with realistic radio effects. In: Proc 3rd ACM conference on embedded networked sensor systems (SenSys'05), San Diego, California, USA, Nov. ACM Press, New York, pp 142–153
19. Tyrrell A, Auer G, Bettstetter C (2006) Fireflies as role models for synchronization in ad hoc networks. In: Proc of 1st bio-inspired models of network, information, and computing systems 2006 (BIONETICS), Madonna di Campiglio, Dec, pp 1–7
20. van Vreeswijk C, Abbott LF (1993) Self-sustained firing in populations of integrate-and-fire neurons. SIAM J Appl Math 53(1):253–264
21. van Vreeswijk C (1996) Partial synchronization in populations of pulse-coupled oscillators. Phys Rev E 54(5):5522–5537
22. Ernst U, Pawelzik K, Geisel T (1995) Synchronization induced by temporal delays in pulse-coupled oscillators. Phys Rev Lett 74(9):1570–1573
23. Ernst U, Pawelzik K, Geisel T (1998) Delay-induced multistable synchronization of biological oscillators. Phys Rev E 57(2):2150–2162
24. Niu Y, d'Auriol BJ, Lee YK, Lee SY (2009) An analysis method for dynamical system. In: Proceedings of the 32th Korea information processing society (KIPS) fall conference, Seoul, Korea, Nov, pp 45–P2–086
25. Hong YW, Scaglione A (2003) Time synchronization with pulse-coupled oscillators for UWB wireless ad hoc networks. In: Proc IEEE conference on ultra wideband systems and technologies, Reston, Virginia, USA, Nov, pp 190–194
26. 802.15.4a-2007 (2007) IEEE standard for local and metropolitan area networks—specific requirements—Part 15.4: Wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (WPANs): Amendment 1: Add alternate phys." IEEE, Tech Rep, Aug 2007, amendment to IEEE Std 802.15.4-2006. [Online]. Available: http://standards.ieee.org/findstds/standard/802.15.4a-2007.html

27. Arslan H, Chen ZN, Benedetto M-GD (2006) Ultra wideband wireless communication. Wiley-InterScience, Hoboken, Chap 14, pp 318–339
28. Benedetto M-GD, Nardis LD, Junk M, Giancola G (2005) (UWB)2: Uncoordinated, wireless, baseborn medium access for UWB communication networks. Mob Netw Appl 10(5):663–674
29. d'Auriol BJ, Niu Y, Lee S, Lee Y-K (2009) The plasma free space optical model for ubiquitous systems. In: Proceedings of the 3rd international conference on ubiquitous information management and communication (ICUIMC-09), Sungkyunkwan University, Suwon, Korea, Jan. ACM Press, New York, pp 446–455
30. IEEE 802.11g-2003 (2003) Further higher data rate extension in the 2.4 GHz band (PDF), IEEE, Tech Rep, O ct retrieved 2007-09-24. [Online]. Available: http://standards.ieee.org/getieee802/download/802.11g-2003.pdf
31. Ghosh AK, Kunta S, Verma P, Huck RC (2010) Free-space optics based sensor network design using angle-diversity photodiode arrays. In: Proceedings of SPIE, San Diego, CA, USA, Aug, vol 7814, p 78140U
32. Verma P, Ghosh AK, Venugopalan A (2008) Free-space optics based wireless sensor network design. In: 1st Symposium on military communications, Prague, Czech Republic, April
33. Nakano H, Utani A, Miyauchi A, Yamamoto H (2008) Synchronization-based data gathering scheme using chaotic pulse-coupled neural networks in wireless sensor networks. In: IEEE International joint conference on neural networks, Hong Kong, Jun 2008. IEEE Press, New York, pp 1115–1121
34. Kusy B, Maroti M (2004) Flooding time synchronization in wireless sensor networks. In: Proc 2nd ACM conference on embedded networked sensor systems (SenSys'04), Baltimore, Maryland, USA, Nov., ACM Press, New York, pp 39–49
35. Lindesey S, Raghavendra C (2002) Pegasis: power-efficient gathering in sensor information systems. In: Aerospace conference proceedings, 2002, Montana, USA, Mar. IEEE Press, New York, pp 3-1125–3-1130
36. Heinzelman W, Chandrakasan A, Balakrishnan H (2002) An application-specific protocol architecture for wireless microsensor networks. IEEE Trans Wirel Commun 1(4):660–670