A Calculation Method of Closeness Centrality for High Density Wireless Sensor Networks

Shuhrat Dehkanov*, Young-rag Kim*, Bok-man Lee*, Chong-gun Kim**
*Graduate School of Computer Engineering, YeungNam University
**Dept. of Computer Engineering, YeungNam University, Korea

Abstract

Centrality has been actively studied in network analysis field. In this paper we show a calculation method of closeness centrality for WSN. Since nodes in a sensor network are very scarce in energy and computation capability the calculation of the closeness is done in two tiers by dividing network into clusters. In first step closeness centrality for cluster heads is calculated. In the second step closeness of member nodes of the chosen cluster is computed in respect to that cluster itself.

Keywords: Wireless sensor networks, closeness centrality, clustering, node density

1. Introduction

Recent technological developments have made it possible to use very tiny sensors in environment monitoring, surveillance applications, military services and many other areas. The main constraint of these sensors is their limited energy capacity. A network consist of such sensors is usually deployed in remote area and thus recharging sensor nodes is not an option. Thus energy conservation is one of the main aspects of research in wireless sensor network field. Many contemporary protocols have been proposed for this purpose.

Applicability of centrality algorithms in such networks is very restricted due to complicacy of the algorithms and above mentioned characteristics of wireless sensor networks. On one hand, centrality measures can be computed by a base station and disseminated to the network. But this process is time consuming in the sense of gathering/disseminating network information by the base station and requires central authority of the base station. Consequently that eliminates self-organized characteristics of the wireless sensor networks.

However, owing to the clustered nature of WSN centrality can still be implemented in these networks. In this paper we introduce two calculation methods of closeness centrality quantity for clustered wireless sensor networks.

Preliminaries:

a) nodes are deployed randomly and they set up multi-hop clustered sensor network
b) once deployed nodes are static, i.e. have no mobility
c) nodes communicate only with their direct neighbors
d) the topology of the network is represented by a graph $G = (V,E)$, where V is a set of vertices, and E is a set of edges
e) the graph is unweighted, undirected and simple
f) distance $d(s, t)$ between two vertices $s,t \in V$ is the number of hops the shortest path connecting them.

The rest of the paper is formulated as follows. First we talk about previous work done on clustered and node density based protocols for sensor networks, and centrality measures. Then we explain our proposed method in detail. We complete the paper with our brief conclusions.

2. Related Works

Various protocols for WSN have been discussed in the literature previously. A typical wireless sensor network consists of vast number of normal nodes, small number of cluster heads (CH) and a base station (BS) (Figure 1).

A well-known clustering protocol is LEACH in which number of sensor nodes become cluster heads with some probability and other nodes join them to make up clusters [3]. Another clustering algorithm is

** Corresponding Author : Chong-gun Kim

HEED – a distributed clustering in ad-hoc sensor networks in which cluster heads are randomly selected based on their residual energy [4]. HEED operates in quasi-stationary networks where nodes are location-unaware and have equal significance.

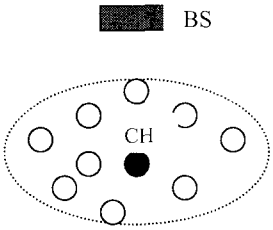

Figure 1. A typical clustered WSN.

Several researches have been conducted on density relation in wireless and ad-hoc sensor networks. Y. Kim et.al [5] has introduced a node density based routing protocol in which transmission paths are directed according to density level of the specific area. In [6] authors have proposed a coverage preserved self-scheduling protocols based on cooperative sensing model in WSN in which nodes are chosen to hibernate in each round when they have minimum contribution to network coverage compared with neighbors in order to conserve overall energy dissipation of the network.

2.1. Closeness Centrality Measures

The closeness centrality of node i is the mean geodesic distance from node i to every other node. Closeness centrality is lower for nodes that are more central in the sense of having a shorter network distance on average to other nodes. A commonly used definition for closeness is as follows:

$$Cc(s) = \frac{1}{\sum_{s,t \in V} \frac{1}{d(s,t)}} \qquad (1)$$

In the equation (1) $d(s,t)$ is distance (number of hops) between vertices (nodes) s and t and $1/d(s,t)$ is the closeness between them. The closeness of a node is sum of closeness to all other nodes in the network.

Assume a network with 4 nodes as shown in Figure 2. The closeness centrality of nodes a, b, c and d are 6/11, 0.4, 0.4 and 6/11 respectively:

$$Cc(a) = Cc(d) = \frac{1}{\frac{1}{1} + \frac{1}{2} + \frac{1}{3}} = \frac{6}{11}$$

$$Cc(b) = Cc(c) = \frac{1}{\frac{1}{1} + \frac{1}{1} + \frac{1}{2}} = \frac{2}{5}$$

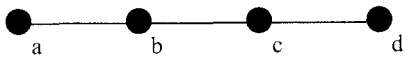

Figure 2. An example network

Accordingly, closeness centrality is lower for nodes that have shorter network distance on average to other nodes.

3. Closeness Centrality for Clustered WSN

The calculation of closeness centrality for wireless sensor networks has $O(n)$ time complexity where n is the number of nodes in the network. Hence, based on the preliminaries given in the introduction and densely deployment phenomenon we introduce two methods for computing closeness centrality in WSN.

Consider a network of 144 nodes uniquely distributed along the sensed area (Figure 3). For the sake of simplicity we assume that diagonal nodes have no direct connection while computing closeness centrality. Definitely closeness centrality of the nodes marked with black is the lowest in this network. According to conventional method of calculation finding closeness centrality of these nodes has $O(144)$ time complexity.

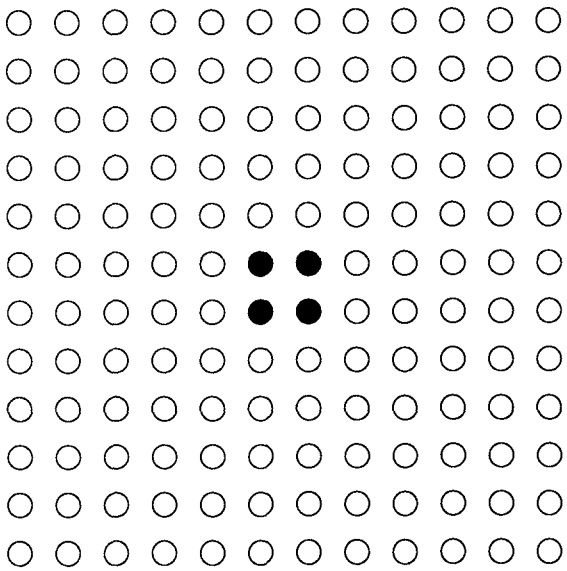

Figure 3. A network with 144 nodes

Method 1. After computing closeness centrality for all nodes, in order to find the nodes with lowest value all the quantities should be compared with each other. This requires higher level of computation capability. Instead sum of closeness centrality for each cluster can be computed and the compared with each other. This way a cluster with lowest sum of closeness is selected.

$$Cc(k) = \sum \frac{1}{\sum_{s,t \in K} d(s,t)} \qquad (2)$$

Sum of closeness centralities, *Cc(k)*, for the member nodes of a cluster is calculated based on the above formula (2) where K is set of nodes of that cluster. Afterwards closeness centrality for the member nodes of that cluster is compared which, on other hand, does not require much memory.

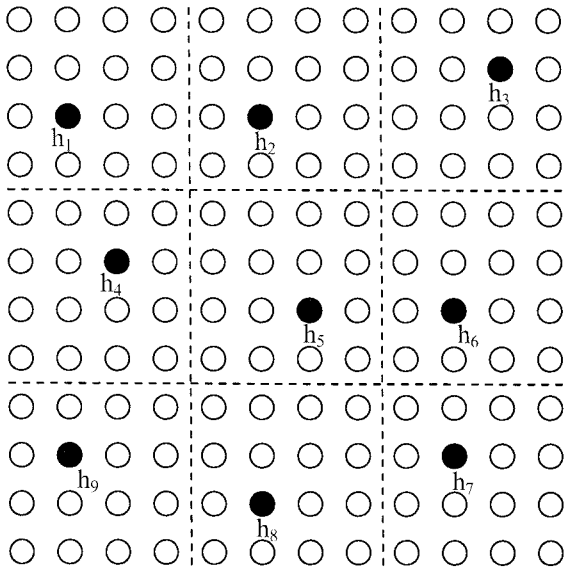

Figure 4. The network with cluster grids nodes

Method 2. Although first method reduces memory requirement for the comparison of the closeness centrality, it still consumes that much time and energy as conventional method. In order to reduce this complexity two steps are introduced.

First, closeness centrality of the cluster heads is computed and the one with lowest value is identified. Assume there are no nodes in the network but only cluster heads (Figure 4), so we will calculate closeness centrality for the network of cluster heads. For the visualization purpose let's first construct shortest path table for cluster heads (Table 1). Please note that we still do not allow direct communication between diagonally positioned nodes. The values in the table cells refer to number of hops between cluster heads, for e.g. $h_1 \rightarrow h_2$: 4 hops, $h_5 \rightarrow h_6$: 3 hops, $h_6 \rightarrow h_8$: 8 hops, etc. Closeness centrality of cluster heads is calculated according to the following equation:

$$Cc(h_i) = \frac{1}{\sum_{\substack{i,j=1 \\ i \neq j}}^{9} \frac{1}{d(h_i, h_j)}} \qquad (3)$$

Note that we do not allow inclusion of distance to particular cluster head itself, $i \neq j$.

Table 1. Shortest path table for cluster

	h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9
h_1		4	10	4	9	12	15	12	7
h_2	4		6	6	5	8	11	8	11
h_3	10	6		12	9	6	9	14	17
h_4	4	6	12		5	8	11	8	5
h_5	9	5	9	5		3	6	5	8
h_6	12	8	6	8	3		3	8	11
h_7	15	11	9	11	6	3		5	8
h_8	12	8	14	8	5	8	5		5
h_9	7	11	17	5	8	11	8	5	

$$Cc(h_1) = \frac{1}{\frac{1}{4}+\frac{1}{10}+\frac{1}{4}+\frac{1}{9}+\frac{1}{12}+\frac{1}{15}+\frac{1}{12}+\frac{1}{7}} = \frac{1}{\frac{1370}{1260}} = \frac{1260}{1370}$$

This way based on the formula (3) and Table 1 the computed closeness centrality for all cluster heads is as follows:

Table 2. Closeness centrality for cluster

$Cc(h_1) = 0.922$	$Cc(h_2) = 0.823$	$Cc(h_3) = 1.151$
$Cc(h_4) = 0.806$	$Cc(h_5) = 0.691$	$Cc(h_6) = 0.723$
$Cc(h_7) = 0.844$	$Cc(h_8) = 0.885$	$Cc(h_9) = 0.968$

A simple comparison of the closeness centrality for cluster heads gives h_5 as the most central compared to others. Thus the cluster five is chosen for the second stage of calculations.

Second, closeness centrality of member nodes for chosen cluster is calculated in relation to that of cluster only. In other words closeness is not calculated in correlation with nodes for whole network but in respect to only that of the chosen cluster. Figure 5 illustrates the cluster of which h_5 is a member. The values below nodes represent the node number.

Table 3 summarizes shortest paths in the cluster in which diagonally located nodes are not allowed to communicate. As in the first step closeness centrality for all nodes is calculated which is shown in Table 4.

Table 4. Closeness centrality for cluster five

n_1	0.163	n_5	0.137	n_9	0.137	n_{12}	0.163
n_2	0.137	n_6	0.117	n_{10}	0.117	n_{13}	0.137
n_3	0.137	n_7	0.117	n_{11}	0.117	n_{14}	0.137
n_4	0.163	n_8	0.132	n_{12}	0.137	n_{15}	0.163

From Table 4 it can clearly be seen that nodes n_6, n_7, n_{10} and n_{11} are the have the lowest closeness value in the cluster. These are the same nodes that are found based on Method 1. This way the time complexity for identifying nodes with lowest closeness centrality can be reduced to $O(k + n/k)$, where k is the number of clusters in the network.

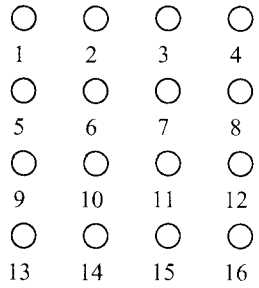

Figure 5. Cluster of h_5

4. Conclusions

In this paper we presented methods of calculating closeness centrality and its applicability for choosing central nodes of a high density wireless sensor network. Our future work will consist of simulation and real test-bed implementation of the proposed method.

5. Acknowledgement

This research was financially supported by the Ministry of Commerce, Industry and Energy (MOCIE) and Korea Industrial Technology Foundation (KOTEF) through the Human Resource Training Project for Regional Innovation.

6. References

[1] K. Akkaya, M. Younis, "A survey on routing protocols for wireless sensor networks", Ad Hoc Networks, Volume 3, Issue 3, pp. 325-349, May 2005.
[2] C. Dangalchev, "Residual closeness in networks", Physica A 365, pp. 556-564, Elsevier 2006.
[3] W. R. Heinzelman, et.al, "An Application-Specific Protocol Architecture for Wireless Microsensor Networks," IEEE Transactions on Wireless Communications,vol.1, no. 4, pp. 660–670, Oct. 2002.
[4] O. Younis, S. Fahmy, "HEED: A Hybrid, Energy-Efficient, Distributed Clustering Approach for Ad Hoc Sensor Networks," IEEE Transactions on Mobile Computing, vol. 3, no. 4, pp. 366-379, Oct., 2004.
[5] Y. Kim, S.H. Kim, C. H. Park, C. Kim, "A Node Density based Routing Algorithm for the Robust Route in MANET," APIS 2008, pp. 97-100, Jan. 2008.
[6] B. Yang, H. Yu, H. Li, H. Hou, "A Coverage-Preserving Density Control Algorithm Based-on Cooperation in Wireless Sensor Networks", WiCom 2006, pp. 1-4, Sept. 2006

Table 3. Shortest path table for cluster five

	n_1	n_2	n_3	n_4	n_5	n_6	n_7	n_8	n_9	n_{10}	n_{11}	n_{12}	n_{13}	n_{14}	n_{15}	n_{16}
n_1		1	2	3	1	2	3	4	2	3	4	5	3	4	5	6
n_2	1		1	2	2	1	2	3	3	2	3	4	4	3	4	5
n_3	2	1		1	3	2	3	2	4	3	2	3	5	4	3	4
n_4	3	2	1		4	3	2	1	5	4	3	2	6	5	4	3
n_5	1	2	3	4		1	2	3	1	2	3	4	2	3	4	5
n_6	2	1	2	3	1		1	2	2	1	2	3	3	2	3	4
n_7	3	2	1	2	2	1		1	3	2	1	2	4	3	2	3
n_8	4	3	2	1	3	2	1		4	3	2	1	5	4	3	2
n_9	2	3	4	5	1	2	3	4		1	2	3	1	2	3	4
n_{10}	3	2	3	4	2	1	2	3	1		1	2	2	1	2	3
n_{11}	4	3	2	3	3	2	1	2	2	1		1	3	2	1	2
n_{12}	5	4	3	2	4	3	2	1	3	2	1		4	3	2	1
n_{13}	3	4	5	6	2	3	4	4	1	2	3	4		1	2	3
n_{14}	4	3	4	5	3	2	3	3	2	1	2	3	1		1	2
n_{15}	5	4	3	4	4	3	2	2	3	2	1	2	2	1		1
n_{16}	6	5	4	3	5	4	3	2	4	3	2	1	3	2	1	