WCCI 2012 IEEE World Congress on Computational Intelligence
June, 10-15, 2012 - Brisbane, Australia IJCNN

QK-Means: A Clustering Technique Based on Community Detection and K-Means for Deployment of Cluster Head Nodes

Leonardo N. Ferreira, A. R. Pinto and Liang Zhao, *Senior Member, IEEE*

Abstract—**Wireless Sensor Networks (WSN) are a special kind of ad-hoc networks that is usually deployed in a monitoring field in order to detect some physical phenomenon. Due to the low dependability of individual nodes, small radio coverage and large areas to be monitored, the organization of nodes in small clusters is generally used. Moreover, a large number of WSN nodes is usually deployed in the monitoring area to increase WSN dependability. Therefore, the best cluster head positioning is a desirable characteristic in a WSN. In this paper, we propose a hybrid clustering algorithm based on community detection in complex networks and traditional K-means clustering technique: the QK-Means algorithm. Simulation results show that QK-Means detect communities and sub-communities thus lost message rate is decreased and WSN coverage is increased.**

I. Introduction

Wireless Sensor Networks (WSN) are composed of small communicating nodes that contain a sensing unit, wireless communication module, processor, memory and a power supply, typically a battery [1], [2]. The set of nodes can be composed by same sensors or some of them may have special characteristics, like different kinds of sensors. These nodes are able to collect data and communicate to each other forming a network with three main kind of topologies: star (one hop), mesh (a routing algorithm between nodes is considered) and cluster-tree (network is divided in clusters) [2].

Some approaches consider a large number of nodes (a dense network), which are deployed near the phenomenon that needs to be monitored. The strategy behind the deployment of a large number of cheap non-reliable nodes has several advantages: (i) better fault tolerance through distributed operation; (ii) uniform covering of the monitored environment; (iii) easy deployment; (iv) reduced energy consumption; and (v) longer network lifetime.

WSN with cluster-tree topology commonly use a special sensor node, called cluster head, that collects the data of each cluster in order to increase coverage and decrease lost messages. Thus, one problem of these networks is the process of cluster detection in order to deploy cluster head sensors. Moreover, when a large number of nodes is used, the connections between these nodes usually create a non-trivial topology resulting in a complex network. In these networks the clustering process, called community detection, becomes a more difficult task.

Complex networks is a young scientific field motivated by observations in real networks. These networks have a non-trivial topology that is different from regular graphs or random graphs but usually are observed in real graphs. Very often, community structures may be observed in these networks. A community is a subset of nodes with a high number of connections between them and a few connections between other nodes. These structures may reveal some information about the dealing network.

In this paper, we propose a hybrid clustering algorithm based on community detection in complex networks and traditional K-means clustering technique, called QK-Means. This new approach takes advantage of both techniques in order to detect better clusters and allow a better deployment of cluster head nodes in large networks. We also propose a network model to simulate the dynamics in a real WSN to make possible the comparison of different approaches. Simulation results show that QK-Means detect communities and sub-communities and, therefore, the lost message rate is decreased and WSN coverage is increased.

The remainder of this paper is organized as follows. First, we present some related works in Section II. Then, in section III are presented a brief introduction of community detection in complex networks. In section IV we describe the networks simulation model and the proposed QK-Means algorithm. In section V are presented and discussed the results of comparisons between our proposed technique and others. Finally, we enumerate some conclusions and future works in Section VI.

II. Related Works

Clustering algorithms are necessary to organize large WSN in clusters in order to decrease the number of lost messages and increase the network coverage. In this section, we present some related clustering approaches for WSN.

Linked Cluster Algorithm (LCA) aims the mobility support of nodes. The main goal of LCA is the formation of a efficient network topology where CHs are hoped to form a backbone network. The cluster members can communicate with this backbone infrastructure while they are moving. LCA assumes that nodes are synchronized and that a time-based medium access is used. Finally, the main LCA objective is the maximization of network connectivity [3].

Leonardo N. Ferreira and Liang Zhao are with the Institute of Mathematics and Computer Science, University of São Paulo, Av. Trabalhador São-carlense 400, Caixa Postal: 668, CEP: 13560-970, Sao Carlos, São Paulo, Brazil (email: leoferr, zhao@icmc.usp.br)

A. R. Pinto is with the DCCE, IBILCE, Universidade Estadual Paulista, UNESP, São José do Rio Preto, SP, Brazil (email: arpinto@ibilce.unesp.br,)

Random Competition based Clustering (RCC) also focus on node mobility and tries to stabilize the formed clusters. RCC applies the first declaration wins rule, this rule considers that the first node that claim being a CH will be elected. After receiving the message of the first node (that is broadcasted) all neighboring nodes join its respective cluster. The CH periodically broadcast a claim packet [4].

The Low Energy Adaptive Clustering Hierarchy (Leach) [5] is one of the most cited approach for WSNs [6]. The WSN clusters are formed based on the received signal strength (RSSI) and a cluster-tree topology is formed (CH are used to route data to the master node). The cluster formation is made in a distributed way, thus nodes autonomously decide which cluster they will belong. The nodes decide if they will be a CH based on a probability p and broadcast its decision, after this step, all non-CH nodes decide its cluster based on the least communication energy. After a predetermined period, the CH are changed in order to save energy and extend WSNs lifetime.

Fast Local Clustering Service (FLOC) is a distributed technique that tries to form equal-sized cluster with minimum overlap [7]. The non-CH nodes are classified based on their proximity to CH into inner (I-band) and outer (o-band). Nodes classified as I-band will suffer low interference during the wireless communication with the CH, and o-band nodes may lose most of their messages due to high interference level. The I-band membership is preferred in order to increase intra-cluster traffic.

CLUBS is an approach that form cluster through local broadcast, cluster are formed with a maximum of two hops. CLUBS is based on the following assumptions: every node must be connected to a cluster, the diameter of all clusters must be the same and the nodes in a cluster must be able to communicate with each others [8].

GS approach [9] is a clustering technique that self-organizes the WSN in a cellular hexagon structure. The main idea is to consider the geographical boundary of clusters, thus the radius of the circle is a measure for the geometric cluster size.

III. Community Detection in Complex Networks

The increasing interest in studying and understanding real networks is motivated by many science fields. An example of these networks is social network [10], where people are represented by nodes and their friendship represented by connections. The World Wide Web is another example, where each webpage is represented by a node and its hyperlinks denoted by edges between nodes [11]. Other examples are energy transmission [12], neural networks [13] and protein iteration [14].

An interesting property observed in many real networks is the modular structure. In these networks, there are a lot of edges between nodes of the same subgroups and few edges between nodes from different subgroups. These groups of vertices, also defined as *communities* [15], are illustrated in Figure 2. A real example of network is the World Wide Web that has many hyperlinks between related web pages and a few hyperlinks between unrelated web pages. The understanding of community structure present in complex networks is interesting to many research fields because it may reveal important information about the dealing problem.

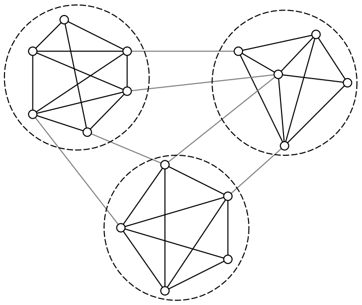


Fig. 1. Network with three communities represented by dotted line circles.

In order to understand and extract information of real networks, many algorithms of community detection was developed. These algorithms are classified as agglomerative ("bottom-up") or divisive ("top-down"). Agglomerative algorithms start considering that every node is a community and merge them forming larger communities. On the other hand, divisive algorithms detect communities by considering at the beginning that the whole network as a community and divide it in smaller communities.

Many techniques have been developed for community detection. A divisive technique calculates minimum path between all nodes, counts the number that every edge was used (edge betweenness) and removes the most used ones [16]. Another method to find communities uses the concept of a Brownian motion developed in [17] and later extended in [18]. A Brownian particle measures the distance between two nodes and it is used to calculate a dissimilarity index. According to this index, network is decomposed into communities. An agglomerative algorithm uses a measure called modularity and merges edges that cause the highest increase of this measure [19]. Another algorithm places the nodes in a circle and move them until the nodes form groups along the circle representing the communities [20].

The agglomerative algorithm proposed by [21] uses the modularity Q that measures the quality of some network division. This algorithm maintains three data structures:

1) A matrix containing ΔQ_{ij} for each pair i e j of communities that has at least one edge between them.
2) A max-heap H with the largest elements of each row of the matrix ΔQ_{ij}
3) An array with elements a_i with the fraction of ends of edges that are attached to vertices in community i.

These structures (ΔQ_{ij} and a array) are initially set according to 1 and 2.

$$\Delta Q_{ij} = \begin{cases} \frac{1}{2m} - \frac{kikj}{(2m)^2} & \text{if } i,\, j \text{ are connected,} \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

$$a_i = \frac{k_i}{2m} \tag{2}$$

where k_i is the degree of a vertice v_i, i.e., the number of edges incident with this node and m the total number of connections (or links, edges) between two different nodes of the network. After setting the data structures, the process of community detection is described in the following algorithm:

```
procedure COMMUNITY DETECTION
    Calculate the initial values of $\Delta Q_{ij}$ and $a_i$ according
        to 1 and 2 respectively;
    Put the largest element of each row of the matrix $\Delta Q_{ij}$
        in max-heap;
    repeat
        Select the largest $\Delta Q_{ij}$ from $H$;
        Join communities $i$ and $j$;
        Update matrix $\Delta Q$, max-heap $H$ and $a_i$;
        $Q \leftarrow Q + \Delta Q_{ij}$;
    until until only one community remains
end procedure
```

When communities i and j are merged, we label this new community as j, update every k element of jth row and column, and remove the ith row and column. To update the matrix ΔQ_{ij} is considered three cases. If community k is connected to both i and j, then

$$\Delta Q'_{jk} = \Delta Q_{ik} + \Delta Q_{jk} \tag{3}$$

If k is connected to i but not j, then

$$\Delta Q'_{jk} = \Delta Q_{ik} - 2a_j a_k \tag{4}$$

If k is connected to j but not i, then

$$\Delta Q'_{jk} = \Delta Q_{jk} - 2a_i a_k \tag{5}$$

Finally update array $a'_j = a_j + a_i$ and $a_i = 0$. For each iteration of the algorithm, Q is incremented with the largest: $Q = Q + \Delta Q_{ij}$. The best network division occurs when Q stops to increase.

IV. Clustering Technique

In this section, we present the model used to simulate a WSN and the clustering technique QK-Means.

A. *WSN Simulation Model*

The used communication model considers one *master node* (base station), CH_n *cluster head* (CH) nodes and S_n *slave* nodes. The data collected by slaves is sent to their respective cluster head nodes that perform the data fusion. All the slave nodes reach the cluster head using just one hop. After a sensing slot time window t, cluster head nodes send their messages to the master node. Slave and cluster head nodes use the same frequency for wireless communication. We consider that cluster heads communicate with master node in a different radio frequency. This difference in radio frequency avoid interference between slave-CH and CH-master communications.

First, the network is built by deploying S_n slaves in the monitoring area. The slaves covered by the cluster heads antenna range are connected, forming clusters of slaves. When a slaves is covered by more than one CHs, this slave will be connected to all of them. The Figure 2 shows an example of network. For each simulation iteration time t, slave nodes generate message with probability λ, store the message on its buffer and tries to send it to its cluster heads. The wireless transmission medium is shared with all nodes inside a specific cluster. Thus, every cluster head can receive just one slave message by time and the other ones store their messages in buffer. This behavior was reproduced by choosing a random slave s of a cluster head ch that has not an empty buffer, decreasing its buffer (s_b) and increasing this CH buffer (ch_b). The same behavior was reproduced with cluster heads and masters. If a slave generates a message and its buffer is full, then this message is considered lost (LM). The massage also is lost when a CH's buffer is full and a slave send information to it.

The following simulation parameters were considered in our model:

- *Number of slaves nodes (S_n):* A specific number of slave nodes are deployed in the monitoring area.
- *Number of cluster heads nodes (CH_n):* A number of CH are deployed in the monitoring area and connected to slave according to its antenna range.
- *Deployment Area:* Simulation area size to deploy nodes.
- *Cluster head buffer size (CH_b):* CH's message storage capacity.
- *Slave buffer size (S_b):* Slave's message storage capacity.
- *Simulation time (T):* Number of iteration used as simulation stop criterium.
- *Probability of message generation (λ):* Every time iteration every slave node has it probability to generate a message.
- *Cluster head antenna radius (CH_r):* Slaves inside this coverage are connected to this CH.
- *Slave sensor radius (S_r):* Slave sensoring coverage radius. If another slave is inside this coverage, an artificial edge is created that will be used to detect community.

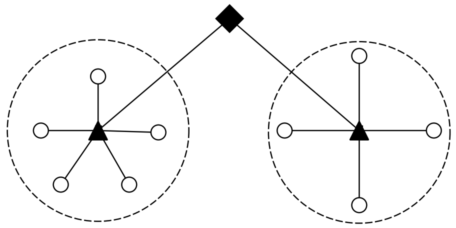


Fig. 2. An example of WSN where slaves are represented by circles, cluster heads by triangles and the master by a diamond. The circles with dotted line show the clusters.

Initially the simulation considers that S_n slave nodes are deployed in a defined monitoring area. After the slave nodes deployment, CH_n cluster head nodes are deployed in the midpoint of the clusters found by some clustering algorithms. Slaves inside CH's antenna range are linked forming clusters.

After the deployment, the monitoring phase starts. Slave nodes begin to generate messages, cluster head nodes receive them and send them to master node. The monitoring phase is described in the following Algorithm:

```
procedure MONITORING PHASE SIMULATION
    $t \leftarrow 0$;
    while $t < T$ do
        for all slaves $s$ do
            if generated a message with probability $\lambda$ then
                if $s_b < S_b$ then
                    $s_b \leftarrow s_b + 1$;
                else
                    $LM \leftarrow LM + 1$;
                end if
            end if
        end for
        for all cluster head $ch$ do
            $s \leftarrow$ random slave from $ch$ with a non-empty
              buffer;
            $s_b \leftarrow s_b - 1$;
            if $ch_b < CH_b$ then
                $ch_b \leftarrow ch_b + 1$;
            else
                $LM \leftarrow LM + 1$;
            end if
        end for
        for all master do
            $ch \leftarrow$ random CH with a non-empty buffer;
            $ch_b \leftarrow ch_b - 1$;
        end for
        $t \leftarrow t + 1$
    end while
end procedure
```

B. QK-Means

Here is proposed a hybrid algorithm to cluster formation based on two approaches: community detection in complex networks and the traditional clustering technique k-means. This new algorithm takes advantage of community detection approach that is able to find clusters of different shapes and K-Means that is a good clustering technique for cartesian points.

The proposed algorithm QK-Means can be divided in three steps: network generation, community detection and find sub-communities:

1) Network Formation: Considering S_n slave nodes deployed in a defined monitoring area, this step consists in creating the network according to the distribution of these slaves. Thus, every slave was considered a node and a link is created between two nodes $v_i \neq v_j$ if $d_{ij} \leq 2CH_r$, i.e., it is possible to deploy a cluster head between them that will be connected to both. On the other hand, if $d_{ij} > 2CH_r$ $\forall$ $v_j \in V$ then it is not possible to deploy a cluster head between the node v_i and another node and therefore no edge will be created. Two examples of networks are illustrated in Figure 3.

2) Community Detection: Once the network was created, the next step consists in detect communities in this network using the algorithm based on modularity, described in Section III.

3) Find Sub-communities: After finding the communities, it is necessary to break this communities into sub-communities that a cluster head can deal with. The K-Means algorithm [22], [23] is used to find these sub-communities. The number of sub-communities $K(C_i)$ in a community C_i with more than one element is calculated (Eq. 6) by the division of the community diameter $Diam(C_i)$ and the cluster head coverage diameter ($2CH_r$). The community diameter is calculated by getting the maximum value returned by Floyd-Warshall algorithm [24]. Communities with just one node are not considered.

$$K(C_i) = \left\lceil \frac{Diam(C_i)}{2CH_r} \right\rceil \tag{6}$$

The following algorithm describes the three steps of the QK-Means algorithm:

```
procedure QK-MEANS
    // Network Formation;
    for all pair of different slaves $v_i$ and $v_j$ do
        if $d_{ij} \leq 2CH_r$ then
            connect nodes $v_i$ and $v_j$;
        end if
    end for
    Detect communities using ;
    // Find sub-communities
    for all community $C_i$ with $n > 1$ do
        $Diam(C_i) \leftarrow max$(Floyd-Warshall($C_i$));
        $K(C_i) \leftarrow \left\lceil \frac{Diam(C_i)}{2CH_r} \right\rceil$;
        K-Means($K(C_i), C_i$);
    end for
end procedure
```

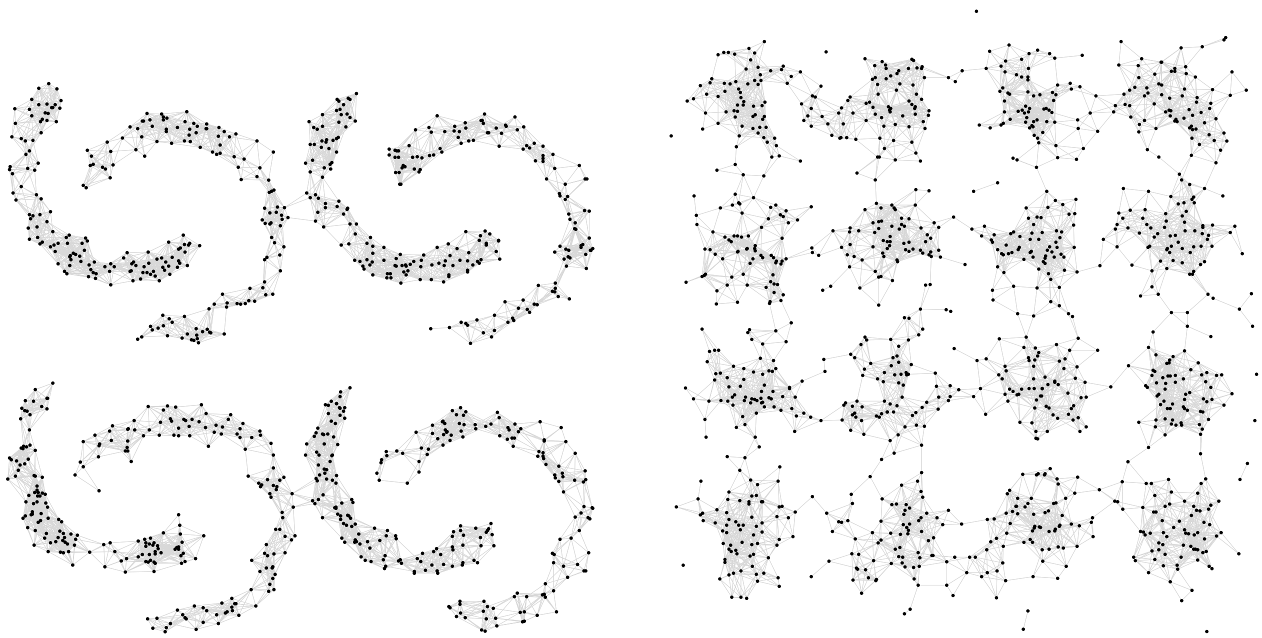
After the clustering process, one cluster head node is deployed in every cluster midpoint.

V. SIMULATION RESULTS

In this section, we discuss about some experiments using QK-Means on the proposed simulation model.

A. Subcluster Size

This first experiment considered the use of different sub-cluster sizes. The proposed QK-Means algorithm uses Equation 6 to find the number of sub-clusters in a community. This experiment consisted of multiplying this equation by some factors, i.e., find smaller sub communities and deploy more cluster head nodes. It was used a pre set network wit 1000 nodes divided in 16 gaussians (Fig. 3(b)). Figure 4 shows the result of these experiments. First, it was used the original equation 6 and then, it was multiplied by 2, 4 and 8. It is possible to observe that as long as K increases, the network coverage increases and lost messages decrease. The coverage



(a)

(b)

Fig. 3. Networks with 1000 slave nodes. (a) 8 Banana-Shaped. (b) 16 Gaussians.

is the percentage of slave nodes that are connected with one or more cluster head nodes. It is important to observe that, in this paper, the main idea of clustering is using the less cluster head nodes to get the higher coverage and therefore get a low rate of lost messages. Thus, it is not interesting the use of a high number of sub-communities.

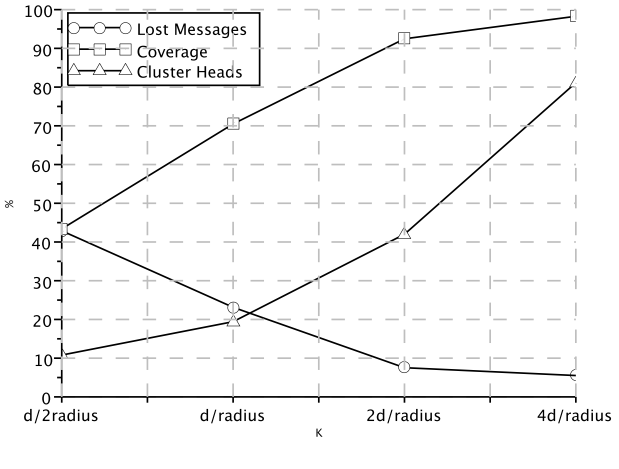


Fig. 4. Effect of the variation of sub-communities number on coverage and lost messages. For this simulations was considered 1000 slave nodes divided in 16 gaussians (Fig. 3(b)). The parameters were the following: $S_r = 50m$, $CH_r = 25m$, $S_b = 50$ messages, $CH_b = 2500$ messages and $\lambda = 10\%$. Each point of graphic represents the mean of 30 simulations.

B. Cluster Head Buffer

In this next experiment was observed the influence of the cluster head buffer (CH_b) on lost message rate. It was used the QK-Means compared to other two algorithm: simple modularity (described in Section III) and expectation-maximization (EM) [25], [26] using a small S_b. Figure 5 shows the results of this simulation. It is possible to notice that as long as the CH buffer increases the lost message rate decreases until reach a point that do not decrease anymore. For this and all simulations was considered that messages on buffer (slave or CH) are not counted as lost messages.

C. Message Generation Probability (λ)

In this experiment was verified the influence of the message generation probability (λ) on lost message rate. Figure 6 shows the variation of lost message when λ increases. As expected, the higher the λ, the higher the lost messages. It is possible to observe that QK-Means had a softer curve than the other algorithms because it deploys more and better positioned cluster head nodes.

VI. Conclusion and Future Works

This paper presented a clustering technique based on community detection in complex networks and traditional clustering algorithm K-Means: the QK-Means. As a hybrid technique, the idea was to take advantage of both approaches. The K-means is a good clustering technique for cartesian points and community detection can find clusters with different shapes in complex networks. An important feature is that QK-Means do not need to know and inform the number of K a priori. The proposed model of wireless sensor networks made

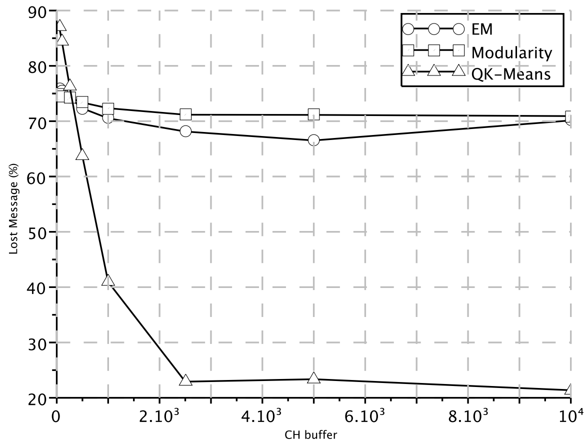


Fig. 5. Variation of cluster head buffer (CH_b) and its influence on lost message rate for some clustering algorithms. For this simulations was considered 1000 slave nodes divided in 8 bananas-shaped (Fig. 3(a)). The parameters were the following: $S_r = 50m, CH_r = 25m, S_b = 50$ messages and $\lambda = 10\%$. Each point of graphic represents the mean of 30 simulations.

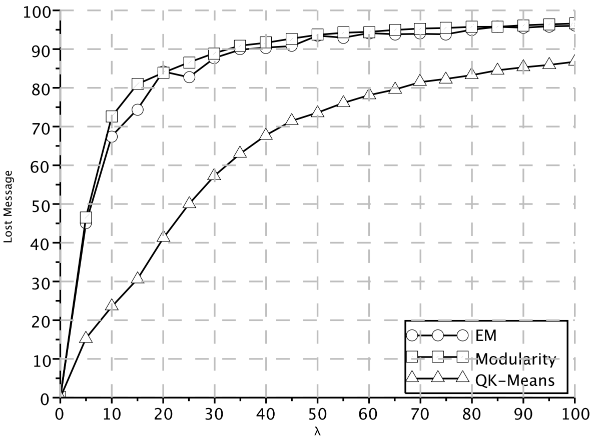


Fig. 6. Variation of message generation probability (λ) and its influence on lost message rate for some clustering algorithms. For this simulations was considered 1000 slave nodes divided in 16 gaussians (Fig. 3(b)). The parameters were the following: $S_r = 50m, CH_r = 25m, S_b = 50$ messages and $CH_b = 2500$ messages. Each point of graphic represents the mean of 30 simulations.

possible the simulation of large networks. It also allowed to make some experiments with QK-Means to check its efficiency.

As future works we could point some improvements in QK-Means like calculating the best CH coverage radius by analyzing the number of nodes in each cluster and if it found a dense cluster, then the algorithm could break them into small clusters and use more CHs with lower CH radius. Try some different algorithms or techniques to calculate community diameter. Try other clustering techniques like DBSCAN or SOM. Use other community detection techniques like Edge Betweenness or Brownian motion. Compare these different combinations of algorithms. Insert parameters to QK-Means like: maximum number of CH or maximum coverage. Create new metrics like the number of delivered messages from slaves to master. Finally, apply QK-Means to other applications by adapting function distance.

Acknowledgment

This paper was supported by National Council for Scientific and Technological Development (CNPq) of Brazil and São Paulo State Research Foundation (FAPESP).

References

[1] I. Akyildiz, “Wireless sensor networks: a survey,” *Computer Networks*, vol. 38, no. 4, pp. 393–422, Mar. 2002.

[2] J. Stankovic, T. Abdelzaher, C. Lu, L. Sha, and J. Hou, “Real-time communication and coordination in embedded sensor networks,” *Proceedings of the IEEE*, vol. 91, no. 7, pp. 1002 – 1022, july 2003.

[3] D. Baker, A. Ephremides, and J. Flynn, “The design and simulation of a mobile radio network with distributed control,” *Selected Areas in Communications, IEEE Journal on*, vol. 2, no. 1, pp. 226 – 237, jan 1984.

[4] K. Xu and M. Gerla, “A heterogeneous routing protocol based on a new stable clustering scheme,” in *MILCOM 2002. Proceedings*, vol. 2, oct. 2002, pp. 838 – 843 vol.2.

[5] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “An application-specific protocol architecture for wireless microsensor networks,” *Wireless Communications, IEEE Transactions on*, vol. 1, no. 4, pp. 660 – 670, oct 2002.

[6] A. A. Abbasi and M. Younis, “A survey on clustering algorithms for wireless sensor networks,” *Computer Communications*, vol. 30, no. 14–15, pp. 2826 – 2841, 2007.

[7] M. Demirbas, A. Arora, and V. Mittal, “Floc: A fast local clustering service for wireless sensor networks,” in *Workshop on Dependability Issues in Wireless Ad Hoc Networks and Sensor Networks (DIWANS/DSN*, 2004.

[8] R. Nagpal and D. Coore, “An algorithm for group formation in an amorphous computer,” in *Proceedings of the 10th International Conference on Parallel and Distributed Computing Systems (PDCS’98), Nevada, USA*, Oct. 1998.

[9] H. Zhang and A. Arora, “Gs3: scalable self-configuration and self-healing in wireless sensor networks,” *Computer Networks*, vol. 43, no. 4, pp. 459 – 480, 2003.

[10] J. P. Scott, *Social Network Analysis: A Handbook*. SAGE Publications, January 2000.

[11] B. A. Huberman, *The Laws of the Web: Patterns in the Ecology of Information*. The MIT Press, October 2001.

[12] L. A. N. Amaral, A. Scala, M. Barthélémy, and H. E. Stanley, “Classes of small-world networks,” *Proceedings of the National Academy of Sciences of the United States of America*, vol. 97, no. 21, pp. 11 149–11 152, October 2000.

[13] O. Sporns, “Networks analysis, complexity, and brain function,” *Complex.*, vol. 8, pp. 56–60, September 2002.

[14] A.-L. Barabási and Z. N. Oltvai, “Network biology: understanding the cell’s functional organization,” *Nature Reviews Genetics*, vol. 5, no. 2, pp. 101–113, February 2004.

[15] M. Girvan and M. E. J. Newman, “Community structure in social and biological networks,” *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, no. 12, pp. 7821–7826, June 2002.

[16] M. E. J. Newman and M. Girvan, “Finding and evaluating community structure in networks,” Aug 2003.

[17] H. Zhou, “Network landscape from a brownian particle’s perspective,” *Phys. Rev. E*, vol. 67, no. 4, p. 041908, Apr 2003.

[18] ——, “Distance, dissimilarity index, and network community structure,” *Physical Review E*, vol. 67, no. 6, pp. 061 901+, Jun. 2003.

[19] M. E. J. Newman, “Fast algorithm for detecting community structure in networks,” *Physical Review E*, vol. 69, no. 6, pp. 066 133+, Jun 2004.

[20] T. de Oliveira, L. Zhao, K. Faceli, and A. de Carvalho, "Data clustering based on complex network community detection," in *Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on*, june 2008, pp. 2121 –2126.

[21] A. Clauset, M. E. J. Newman, and C. Moore, "Finding community structure in very large networks," *Physical Review E*, vol. 70, no. 6, pp. 066 111+, Dec. 2004.

[22] J. B. Macqueen, "Some methods of classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1967, pp. 281–297.

[23] H.-H. Bock, "Clustering Methods: A History of k-Means Algorithms," in *Selected Contributions in Data Analysis and Classification*, ser. Studies in Classification, Data Analysis, and Knowledge Organization, P. Brito, G. Cucumel, P. Bertrand, and F. Carvalho, Eds. Springer Berlin Heidelberg, 2007, ch. 15, pp. 161–172.

[24] R. W. Floyd, "Algorithm 97: Shortest path," *Commun. ACM*, vol. 5, no. 6, pp. 345+, Jun. 1962.

[25] H. O. Hartley, "Maximum Likelihood Estimation from Incomplete Data," *Biometrics*, vol. 14, no. 2, pp. 174–194, 1958.

[26] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.