ISSN:0975-9646

Dynamic Clustering in Sensor Networks using HMM and Neuro Computation

Veena k.N.[#1], Vijaya Kumar B.P.[*2]

[#1] *Department of Telecommunication Engg, J.N.N. College of Engineering, Shimoga, India*

[*2] *Department of Telecommunication Engg., M.S.Ramaiah Institute of Technology, Bangalore, India*

***Abstract*— Most important challenge in Wireless Sensor Networks is to improve the operational efficiency in highly resource constrained environment based on dynamic and unpredictable behavior of network parameters and applications requirement. In this paper we have proposed a technique of clustering and analysis, to study the system behavior with respect to network parameters and applications requirement. The technique involves in the adoption of Fuzzy logic and Hidden Markov Model for the analysis of sensor node parameters and Neuro computation for clustering. The simulations are carried out to evaluate the performance of the proposed method with respect to different parameters of sensor networks and applications requirement.**

***Keywords*— Wireless Sensor Networks, Clustering, Hidden Markov Model, Neural networks, Fuzzy logic.**

I. INTRODUCTION

In the recent years, Wireless Sensor Networks (WSNs) has found a wide variety of applications and systems with vastly varying requirements and characteristics. As a result, the complexities of the system design issues and their requirement have increased with respect to hardware and software. Integrating sensor nodes into sophisticated sensing, computational and communication infrastructures to form wireless sensor networks will have a significant impact on a wide array of applications ranging from military, scientific, industrial, health-care and domestic services. Various applications of WSN are Telemedicine, Habitat monitoring, Structure health monitoring, Active Volcano, Seismic monitoring, and Avalanche Victims. Wireless sensor networks are usually a large number of sensor nodes, which are tiny, compact and low cost embedded devices, which can be readily deployed in various types of unstructured environments within predefined and specified area or sometimes an approximate area of interest, either inside the phenomenon or very close to it. Efficiency of such systems would improve by providing distributed, localized and energy efficient techniques. Here the sensor nodes have native capabilities to detect the nearest neighbors and help to develop an ad-hoc network through a set of well-defined protocols. This leads to the existence of clusters of nodes, which can enhance the network operations under network management, processing and aggregation of sensor data. Therefore there are many advantages with clustering and their analysis based on the system parameters and applications requirement. The systematic clustering by choosing the key parameters and response to their dynamic behavior in resource constrained sensor networking environment will be a challenging and complex issue.

In this work, we have proposed a technique of clustering and analysis to study the dynamic behavior of the system that includes sensor network parameters and applications requirement. This paper is extension of our previous paper[1]. The method involves the Fuzzy Logic in fuzzifyng the sensor node parameters. Hidden Markov Model (HMM) is used in estimating a sensor node suitable for a particular application. We have used Kohonen Self Organization Map Neural Network (KSOM-NN) [2] as Computational Intelligence for clustering.

The organization of the paper is as follows. A brief discussion on some of the related works is explained in section II. Section III explains the proposed Dynamic clustering model. Concept of clustering is explained in section IV. The description of the proposed clustering algorithm is narrated in section V. Simulation and results are described in section VI and concluding remarks are given in section VII.

II. RELATED WORKS

In this section a brief discussion on the related works pertaining to Fuzzy logic, Hidden Markov Model and clustering in WSNs are discussed. In [3] fuzzy logic systems to analyze the lifetime of a wireless sensor network are presented. It uses a type-2 fuzzy membership function (MF), where a Gaussian MF with uncertain standard deviation is used to model a single node lifetime in wireless sensor networks. Clustered WSN implemented in [4], it implements a protocol for trade-off between loads and increase in life expectancy of the network. In [5] a two-level fuzzy logic is utilized to evaluate the qualification of sensors to become a cluster head. In the first level the qualified nodes are selected based on their energy and number of neighbors.. Then, in the second level nodes overall cooperation is considered in the whole network with fuzzy parameters are discussed. A human movement monitoring, which also adapts to the new movement and new sensors using hidden markov model is implemented in [6]. In [7] an action recognition framework based on an HMM, which is capable of both segmenting and classifying continuous movements for the distributed architecture of Body Sensor Networks is presented. In [8] wearable sensor network that monitors relative proximity using Radio Signal Strength indication (RSSI), and then construct a HMM system for posture identification in the presence of sensing are discussed.

A distributed, randomized clustering algorithm [9] to organize the sensors in a wireless sensor network into clusters is discussed. The algorithms generate a hierarchy of cluster heads and observe that the energy savings increase with the number of levels in the hierarchy. The Linked Cluster Algorithm [10] where a node becomes the cluster head, if it has the highest identity among all the nodes within one hop of itself or among its neighbors is presented. Budget-based clustering approaches [11], message efficient distributed clustering [12] are proposed, involving autonomous sensor network clusters of bounded size by keeping lower overall message complexity. In [13] determination of the optimal number of clusters in an observation area is implemented.

III. DYNAMIC CLUSTERING MODEL

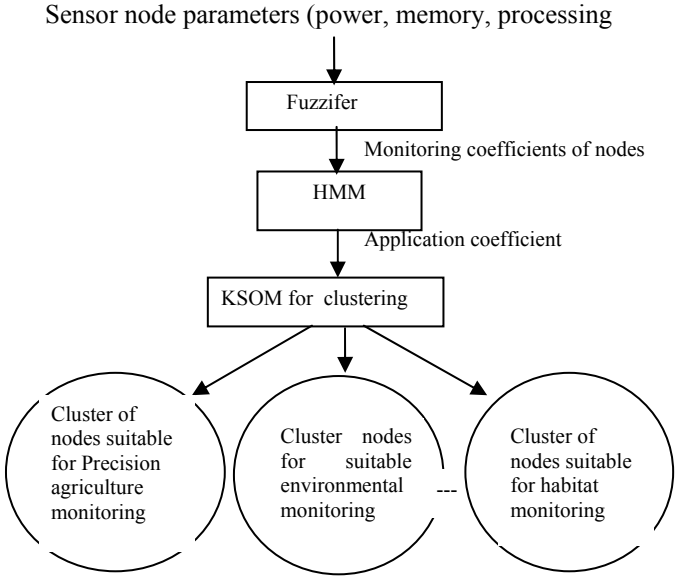

Fig. 1 Dynamic Clustering model

Dynamic clustering model using Neuro-Fuzzy Technique and Hidden Markov Model is shown in fig 1. The sensor data readings are fuzzified. Ex: Power present in sensor nodes, memory available and processing speed of sensor nodes are given to fuzzifier. The fuzzifier assigns monitoring coefficients for each sensor nodes according to predefined fuzzy rules. Ex: Monitoring coefficients for sensor node which can be used for critical monitoring, used for event monitoring, continuous and cannot be used for monitoring because of very low power availability. These coefficients are then given HMM to estimate the application coefficient for each sensor node that is suitable for various applications. Ex: A senor node can be suitable for Precision agriculture, Environmental monitoring, Habitat monitoring, Disaster monitoring etc. These coefficients of sensor node are given to KSOM neural network for clustering. Depending on the application requirement, cluster of sensor nodes suitable for Precision agriculture, environmental monitoring, seismic monitoring, habitat monitoring and disaster monitoring are obtained.

IV. CLUSTERING IN WIRELESS SENSOR NETWORKS

Clustering is the architecture of choice as it keeps the traffic local and sensor nodes would send information only to the nearby cluster head within a fixed radius. Clustering sensor nodes are advantageous because they conserve limited energy resources (power) and improve energy efficiency. In a cluster, each cluster will contain a cluster head. Each cluster head gathers information from its group of sensors, perform data aggregation and relay only relevant information to the sink as shown in fig 2. By aggregating the information from individual sensors, can abstract the characteristics of network topology along with applications requirement and it also reduces the bandwidth. This provides scalability and robustness for the network and increases the lifetime of the system.

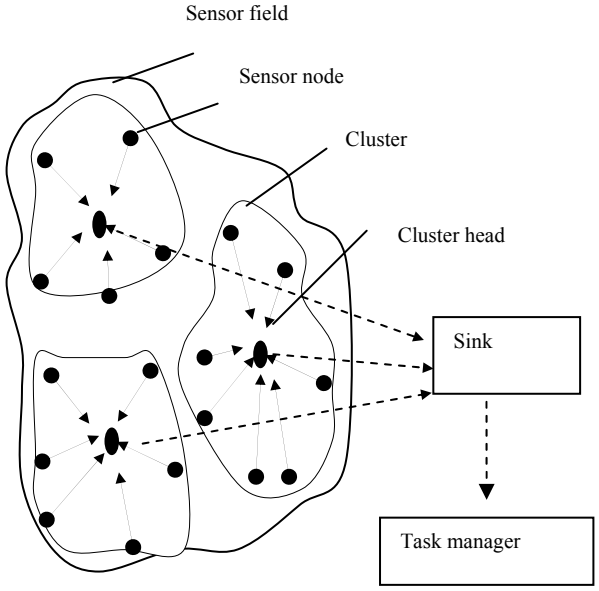

Fig. 2 Clustered Wireless Sensor Network

V. PROPOSED CLUSTERING ALGORITHM

We propose a technique for clustering and cluster analysis to study the behavior and operation of WSN with respect to network parameters and applications requirement. There are application parameters, which will have the influence over the resources available in WSN. Hence there is a need to understand how WSN behaves keeping its resource-constrained environment with respect to applications requirement. For clustering we have adopted the computational intelligence technique, which deals with the numerical data and does not use knowledge in the AI sense and exhibits computational adaptivity with fault tolerance [14]. This study is essential to enhance the clarity in understanding the behavior of WSN parameters pertaining to the specific applications to improve the operational efficiency. The proposed clustering algorithm is given below.

Algorithm 1. Proposed Dynamic Clustering Algorithm
Begin

1. *The sensor node parameters such as memory, power available, processing speed of each sensor nodes is considered.*
2. *These sensor node parameters are fuzzified using fuzzy logic with fuzzy rules as given in Table1.*

3. The fuzzifier assigns monitoring coefficient for each node, depending on its power availability, memory availability, processing speed.

4. These monitoring coefficients are given to HMM to estimate application coeffi cient for each sensor node.

5. These application coefficients are given to KSOM neural network to obtain clusters depending on the application requirement.

End

5.1 Fuzzy Logic

The model using fuzzy logic [15] consists of a fuzzifier, fuzzy rules, fuzzy inference engine, and a defuzzifier. Input variables used are sensor node parameters such as power available in sensor node, memory and processing speed. Membership function and Fuzzy rules are applied to the sensor node parameters. The membership functions developed and their corresponding linguistic states are represented in Table 1 and fig 3 through fig 6. The Fuzzifier assigns the monitoring coefficients for each sensor nodes.

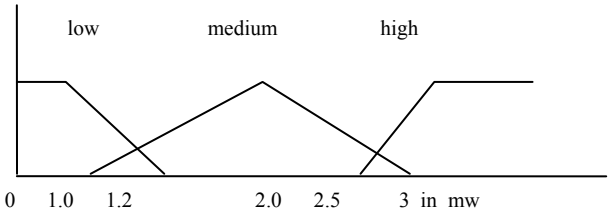

Fig. 3 Membership function for power present in sensor node

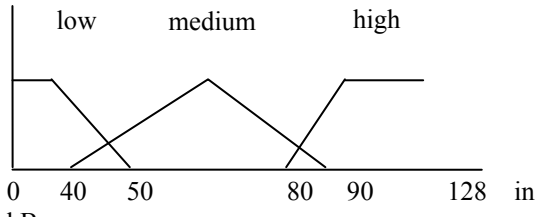

Fig. 4 Membership function for memory available in sensor node

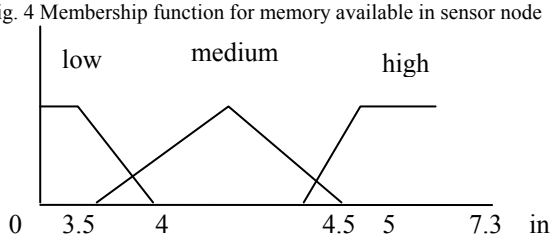

Fig. 5 Membership function for processing speed in sensor node

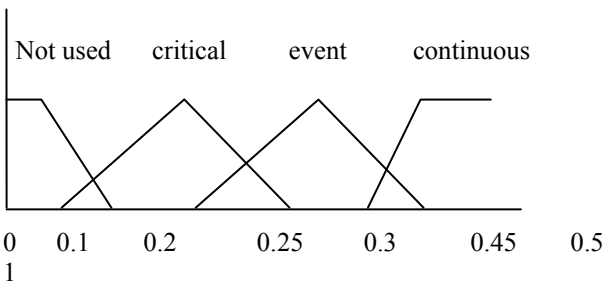

Fig. 6 Membership function for monitoring coefficient of sensor node

TABLE 1 FUZZY RULES

	Power	Memory	Processing speed	Monitoring coefficient
1	low	low	low	Not used
2	medium	medium	medium	critical
3	medium	high	medium	event
4	medium	medium	high	event
5	medium	high	high	event
6	high	high	medium	continuous
7	high	medium	high	continuous
8	high	high	high	continuous

Sensor nodes which have high power, memory and processing speed can be used for continuous monitoring. Sensor nodes with medium power, high memory and high processing speed can be used for event monitoring, sensor nodes with medium power and low memory and processing speed can be used for critical monitoring. Sensor nodes with low power and low memory and low processing speed cannot be used for monitoring.

5.2 Hidden Markov Model

A Hidden Markov Model (HMM)[16] is a statistical Markov model in which the system being modeled is assumed to be a Markov process with unobserved (*hidden*) states. A Hidden Markov Model is shown in the fig 7 and it is denoted by equation 1.

$$\lambda=(\pi,A,B) \quad (1)$$

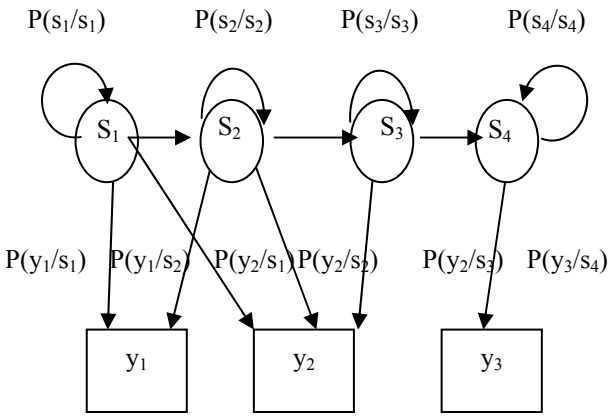

Fig. 7 General Architecture of a Hidden Markov Model

s- states (application coefficients); y-observations (monitoring coefficients); a- State transition probabilities; b-output probabilities;

N applications of WSN are modeled as N hidden states, with $S = \{S_1, S_2, \ldots, S_N\}$ the set of hidden states, in our case N=4. y is the number of observable parameters, y monitoring coefficients of sensor nodes are considered, with $y = \{ y_1, y_2, \ldots, y_T\}$ the set of observable parameters, in our case m=3.

$A=\{a_{ij}\}$, the transition probabilities between the hidden states S_i and S_j, $a_{ij} = P[S_j(t+1)| S_i(t)]$, $0 \le i, j \le N$.

$B = \{b_{jk}\}$ - the probabilities of the observable states y_k in hidden states S_i, $b_j(k) = P[y_k(t)| S_j(t)]$, $0 \le j \le N$, $0 \le k \le T$.

The following transition state matrix probabilities and emission probabilities are considered for HMM computation.

$$A=\begin{bmatrix} 0.8 & 0.2 & 0 & 0 \\ 0 & 0.8 & 0.2 & 0 \\ 0 & 0 & 0.8 & 0.2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad B=\begin{bmatrix} 07 & 0.3 & 0 \\ 0.6 & 0.4 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

π = The initial hidden state probabilities, $\pi_i = P[S_i]$, $0 \le i \le N$.

$$\sum_j a_{ij} = 1 \text{ for all } i, \quad \sum_k a_{jk} = 1 \text{ for all } j$$

5.2.1 Algorithms of HMM

This section introduces the three main algorithms for the solution of the standard problems of HMM is given. They are

The Decoding Problem:

Given a sequence of emissions y^T over time T and a HMM with complete model parameters, meaning that transition and emission probabilities are known, this problem computes the most probable underlying sequence S^T of hidden states that led to this particular observation.

The Evaluation Problem:

Here the HMM is also given with complete model parameters, along with a sequence of emissions y^T. In this problem the probability of a particular y^T to be observed under the given model has to be determined.

The Learning Problem:

The elemental structure of the HMM is given. Given one or more output sequences, this problem computes the model parameters A, B. i.e the parameters of the HMM have to be trained.

Forward Algorithm:

The Forward algorithm is gives the solution for the Evaluation Problem. Forward algorithm and time reversed version of backward algorithm is discussed.

$\alpha_i(t)$ is the probability that the model is in state $S_i(t)$ and has generated the target sequence upto step t. $\beta_i(t)$ is the probability that the model is in state $S_i(t)$ and will generate the remainder of the given target sequence from t+1 to T.

$$\alpha_j(t) = \begin{cases} 0 & t = 0 \text{ and } j \neq \text{initial state} \\ 1 & t = 0 \text{ and } j = \text{initial state} \\ \left(\sum_i \alpha_i(t-1)\, a_{ij}\right) b_{jk}\, y(t) & \text{otherwise} \end{cases}$$

Algorithm 2: Forward Algorithm

Begin

1. *Initialize t=0,* a_{ij}, b_{jk}, *observed sequence* y^T, $\alpha_j(0)$
2. *For* $t \leftarrow t+1$
3. $\alpha_j(t) \leftarrow b_{jk}\, y(t) \sum_{i=1}^{N} \alpha_i(t-1) a_{ij}$
4. *until t=T*
5. *Return* $P(y^T) \leftarrow \alpha_0(T)$ *for the final state*

End

Algorithm 3:Backward Algorithm

Begin

1. *Initialize* $\beta_j(T)$, $t \leftarrow T$, a_{ij}, b_{jk}, *observed sequence* y^T
2. *For* $t \leftarrow t-1$
3. $\beta_j(t) \leftarrow \sum_{j=1}^{N} \beta_j(t+1) a_{ij}\, b_{jk}\, y(t+1)$
4. *until t=1*
5. *Return* $P(y^T) \leftarrow \beta_i(0)$ *for the known initial state*

End

Decoding algorithm:

The Decoding problem is solved by the Decoding algorithm, which is sometimes called Viterbi algorithm.It is used to find the most likely sequence of hidden states to have generated a particular output sequence, given the model parameters.

Algorithm 4: Decoding Algorithm

Begin

1. *Initialize Path* $\leftarrow \{\}$, $t \leftarrow 0$
2. *For* $t \leftarrow t+1$
3. $j \leftarrow j+1$
4. *for* $j \leftarrow j+1$
5. $\alpha_j(t) \leftarrow b_{jk}\, y(t) \sum_{i=1}^{N} \alpha_j(t-1) a_{ij}$
6. *until j=N*
7. $j' \leftarrow \arg\max_j \alpha_j(t)$
8. *Append* $s_{j'}$ *to path*
9. *until t=T*
10. *Return path*

End

Baum-Welch algorithm

The Baum-Welch algorithm, also known as Forward-Backward Algorithm, is used for Learning Problem.

$$\beta_j(t) = \begin{cases} 0 & s_i(t) \neq s_0 \text{ and } t \neq T \\ 1 & s_i(t) = s_0 \text{ and } t \neq T \\ \sum_j \beta_j(t+1) a_{ij}\, b_{jk}\, y(t+1) & \text{otherwis} \end{cases}$$

The probability of transition between $s_i(t-1)$ and $s_j(t)$ give the model generated the entire training sequence y^T by any path is

$$\gamma_{ij}(t) = \frac{\alpha_i(t-1) a_{ij} b_{jk} \beta_j(t)}{P(y^T|\lambda)}$$

Here λ denotes the HMMs model parameters (aij and bjk), and therefore $p(V^T|\lambda)$ is the probability that the model generated V^T. Hence the auxiliary quantity is the probability of a transition from Si(t − 1) to Si(t), under the condition that the model generated V^T.

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T} \gamma_{ij}(t)}{\sum_{t=1}^{T} \sum_{k} \gamma_{ik}(t)} \qquad (2)$$

Numerator is the expected number of transitions between state $S_i(t-1)$ and $S_i(t)$, Denominator is the total expected number of transitions from S_i.

$$\hat{b}_{jk} = \frac{\sum_{t=1, y(t)=y_k}^{T} \sum_{l} \gamma_{jl}(t)}{\sum_{t=1}^{T} \sum_{l} \gamma_{jl}(t)} \qquad (3)$$

It is the ratio between frequency that any particular symbol yk is emitted and that for any symbol.

Algorithm 5: Baum Welch Algorithm

Begin

1. *Initialize* a_{ij} , $b_{ik,}$ *training sequence* y^T, $z \leftarrow 0$, θ *convergence criterion.*
2. *do* $z \leftarrow z+1$
3. *Compute* $\hat{a}(z)$ *from a(z-1) and b(z-1) by eqn. 2*
4. *Compute* $\hat{b}(z)$ *from a(z-1) and b(z-1) by eqn. 3*
5. $a_{ij}(z) \leftarrow \hat{a}_{ij}(z-1)$
6. $b_{jk}(z) \leftarrow \hat{b}_{jk}(z-1)$
7. *Until* $\max_{i,j,k}[a_{ij}(z) - a_{ij}(z-1), b_{jk}(z) - a_{jk}(z-1)] < \theta$
8. *Return* $a_{ij} \leftarrow a_{ij}(z); \quad b_{jk} \leftarrow b_{jk}(z)$

End

5.3 Kohonen Self Organizing Map-Neural Network

Clustering computation is carried out using Kohonen Self–Organizing Map Neural Networks (KSOM-NN) algorithm to cluster the sensor nodes depending on the parameters of interest. The KSOM-NN is competitive, feed forward type and unsupervised training neural network. They have the capability of unsupervised learning and self-organizing properties, is able to infer relationships and learn more as more inputs are presented to it. The Application coefficient of each sensor nodes is considered to cluster sensor nodes using Kohonen Self-Organizing Map Neural Networks. Cluster of sensor nodes suitable for various applications are formed. The fig 8 shows a typical example of a KSOM-NN [17] with 4 input and 20 output neurons.

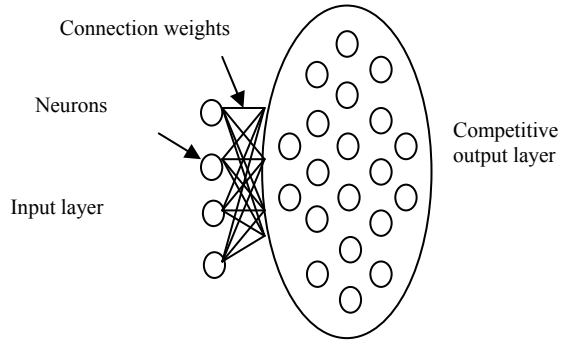

Fig. 8 A typical KSOM-NN model with 4 input and 20 output neurons

Let the input pattern be: $I = (I_1, I_2,.,I_{|V|})$, where there are |V| input neurons in the input layer. Now suppose there are Q neurons in the output (competitive) layer, let O_j be the j^{th} output neuron. So the whole of the output layer will be: $O= (O_1,O_2,.,O_Q)$ for each output neuron O_j there are |V| incoming connections from the |V| input neurons. Each connection has a weight value W. So, for any neuron O_j on the output layer the set of incoming connection weights are: $w_j=(w_{j1},w_{j2},.,w_{j|v|})$. The Euclidean distance value D_j of a neuron O_j in the output layer, whenever an input pattern I is presented at the input layer, is:

$$D_j = \sqrt{\sum_{i=1}^{|V|.} (I_i - W_{ji})^2} \qquad (4)$$

The competitive output neuron with the lowest Euclidean distance at this stage is the closest to the current input pattern, called as winning neuron, O_c. The fig. 9 shows an example of a neighborhood around a winning neuron O_c.

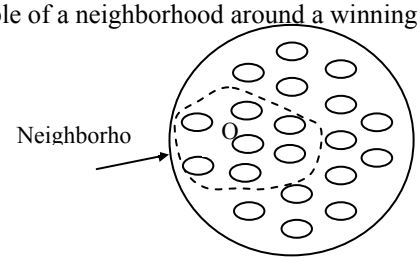

Fig.9 A 4X4 Kohonen grid showing the size of neighborhood influence around neuron O_c.

As shown in figure 3, the neighborhood size and weight updating computation use the following equations (4) and (5). The size of the neighborhood h, starts with a big enough size and decreases with respect to learning iterations such that:

$h_t=h_0(1-t/T)$ ----------(5)

Where, h_t denotes the actual neighborhood size, h_0 denotes the initial neighborhood size, t denotes the current learning epoch and T denotes the total number of epochs to be done. Where an epoch is when the network has swept through all the input patterns of every sensor node once. During the learning stage, the weights of the incoming connections of each competitive neuron in the neighborhood of the winning one are updated, including the winning neuron.

The overall weights update equation is:

$W_j^{new} = W_j^{old} + \alpha (I - W_j^{old})$ ----------(6)

if unit O_j is in the neighborhood of winning unit O_c. Where, α is the learning rate parameter, typical choices are in the range [0.2 . . 0.5]. A learning algorithm I for clustering is summarized as follows.

Algorithm 6: Learning algorithm for Clustering

Begin

1. *Assign random values to the connection weights, W in the range (0, 1);*
2. *Select an input pattern ie., parameters of the sensor nodes.*
3. *Determine the winner output neuron* O_c *using equation (4);*
4. *Perform a learning step affecting all neurons in the neighborhood of* O_c *by updating the connection weights using equation (5); and update* h_t *and continue with step 2 until no noticeable changes in the connection weights are observed for all the input patterns.*
5. *Stop*

End.

VI. SIMULATION AND RESULTS

In order to evaluate the performance of the proposed technique simulation is carried out based on sensor node characteristic features such as power present in sensor nodes, memory available and processing speed. Fuzzy logic is used to fuzzify the sensor node parameters. The fuzzifier gives the membership function coefficients for monitoring. Then the monitoring coefficients are given HMM to estimate the application coefficient for each sensor node. These application coefficients are given to KSOM to obtain dynamic clusters depending on the application requirement.

Clustering of sensor node using KSOM-NN is computed for various numbers of nodes by taking monitoring coefficients of sensor node. The clustering implementation is carried out using C++ programming language by defining suitable classes and structures. Simulation experiments are carried out rigorously by taking different number of nodes in sensor networks, i.e., 100 to 1000. The simulation program is run for many iterations until they converge to stable clusters. A plot of monitoring coefficients for various parameter of a sensor node such as power, memory available and processing speed is shown in fig 10, and fig 11. The result shows that as power, memory and processing speed is high for a sensor node, then monitoring coefficients generated is also high. Such type of sensor node can used for continuous monitoring. If power present in a sensor node is very low, then monitoring coefficient generated is low such sensor nodes cannot be used for monitoring purpose. These results help us in analyzing, whether a node is capable of monitoring a phenomenon or not and how well it can monitor an area of interest.

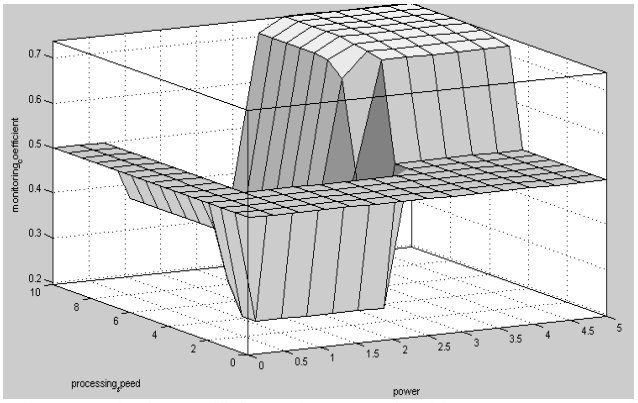

Fig. 10Monitoring coefficients of sensor node with respect to power and processing speed.

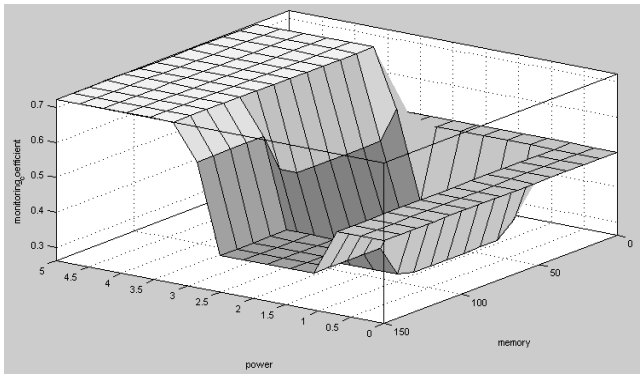

Fig. 11 Monitoring coefficients for sensor node with respect to power and memory

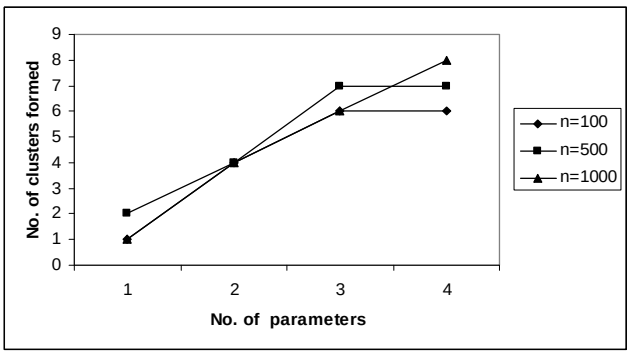

Fig. 12 Cluster formed with respect to application requirement

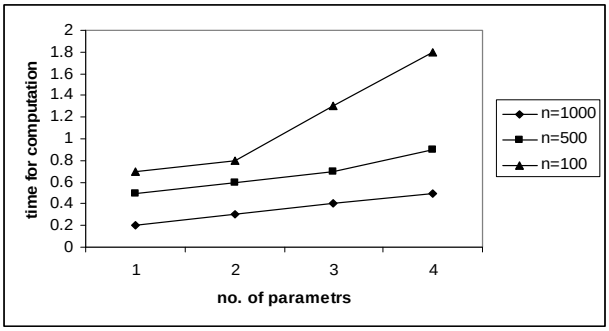

Fig. 13 Time for computation to form clusters with respect to parameters of sensor node

The fig 12 shows the number of clusters formed with respect to application coefficient of sensor nodes in WSN. The results show that the number of clusters formed is dependent of the number of parameters of sensor nodes chosen and the application requirement. These results are essential and can be easily used in various applications depending on the critical region of interest. For example, nodes which can be used for continuous monitoring are suitable for precision Agriculture, and nodes which can be used for event monitoring are suitable for monitoring of seismic area. The computation time for clustering is measured with respect to application and number of sensor nodes as shown in fig 13. These results show on how the individual parameters affect the formation of cluster and its size. These results would be used in real time decision making applications and in adaptive systems.

VII. CONCLUSION

In this work we have proposed a technique for clustering and their analysis to study the dynamic behavior of the cluster formation based on the system parameters and application requirements. The technique involves the adoption of computational intelligence to form clustering and analyzing sensor node parameters in WSN. We have used computational intelligence, Neuro-Fuzzy techniques and Hidden Markov Model for the above computations. The simulations are carried out to evaluate the performance of the proposed method with respect to different parameters of sensor networks and applications requirement. Since we are analyzing and estimating sensor nodes depending on the application requirement, proactive decisions can be taken. The decisions can be redeployment of sensor nodes, when we estimates some of the sensor cannot be used for monitoring because of very low power, or sensor nodes are

less in number for a particular application requirement. This study would give scope for more understanding the chaotic behaviour of environment and system parameters of WSN to enhance the operational efficiency.

REFERENCES

[1] Veena K.N., Vijaya Kumar B.P. "Hidden Markov Model with Computational Intelligence for Dynamic Clustering in Wireless Sensor Networks ", first International Conference on Advances in Computing (ICAdC-2012), MS Ramaiah institute of technology, Bangalore, India, july 4 to 6 2012. Appears in "Advances in Intelligent and Soft Computing" series, Springer

[2] Simon Haykin.: Neural Networks: A Comprehensive foundation, Macmillan college publishing, Newyork, USA, 1995, 2nd edition.

[3] Haining Shu.: Analysis Using Interval Type-2 Fuzzy Logic System, Fuzzy Systems, IEEE Transactions on April 2008 Volume: 16, Issue: 2 On Page(s): 416 – 427.

[4] Islamic Azad, Mashhad, Iran.: CFGA: Clustering Wireless Sensor Network Using Fuzzy Logic and Genetic Algorithm, Wireless Communications, Networking and Mobile Computing (WiCOM), 7th International Conference on 23-25 Sept. 2011 On Pages: 1 – 4.

[5] Torghabeh, N.A.:Cluster head selection using a two-level fuzzy logic in wireless sensor networks, Computer Engineering and Technology (ICCET), 2nd International Conference16-18 April 2010 Volume: 2, On Page(s): V2-357 - V2-361.

[6] Eric Guenterberg Hassan Ghasemzadeh Ruzena Bajcsy.: A Method for Extracting Temporal Parameters Based on Hidden Markov Models in Body Sensor Networks With Inertial Sensors, IEEE Transactions on Information Technology in Biomedicine, VOL. 13, NO. 6, November 2009 page 1019 -1030.

[7] Eric Guenterberg, Hassan Ghasemzadeh, Vitali Loseu, and Roozbeh Jafari.: Distributed Continuous Action Recognition Using a Hidden Markov Model in Body Sensor Networks, BodyNets '08 Proceedings of the ICST 3rd international conference on Body area networks ISBN: 978-963-9799-17-2.

[8] Muhannad Quwaider and Subir Biswas.: Body posture identification using hidden Markov model with a wearable sensor network, BodyNets '08 Proceedings of the ICST 3rd international conference on Body area networks ICST, Brussels, Belgium, Belgium ©2008 ISBN: 978-963-9799-17-2

[9] Seema Bandyopadhyay and Edward J. Coyle.: An Energy Efficient Hierarchical Clustering Algorithm for Wireless Sensor Networks, IEEE INFOCOM, 2003.

[10] D. J. Baker and A. Ephremides.: The Architectural Organization of a Mobile Radio Network via a Distributed Algorithm, IEEE Transactions on Communications, Vol. 29, No. 11, pp. 1694-1701, November 1981.

[11] Leonidas Tzevelekas and Ioannis Stavrakakis.: Directed Budget-Based Clustering for Wireless Sensor Networks, IEEE International Conference, MASS Oct 2006.

[12] R. Krishnan and D. Starobinski.: Efficient clustering algorithms for self organizing wireless sensor networks, Ad Hoc Networks, vol. 4, no. 1, pp. 36.59, January 2006.

[13] Li-Chun Wang, Chuan-Ming Liu, and Chung-Wei Wang.: Optimizing the Number of Clusters in a Wireless Sensor Network Using Cross-layer Analysis, IEEE International Conference, MASS, Oct 2004.

[14] Witold P. and Athanasios Vasitakos.: Computational Intelligence in Telecommunication Networks, CRC press LLC, USA, 2001.

[15] Veena K.N. and Vijaya Kumar B.P " Hidden Markov Model with Computational Intelligence for Dynamic Clustering in Wireless Sensor Networks", International Conference on Advance Computing (ICAdC-2012),M.S.R.I.T, Bangalore, July 4 to 6. Appears in "Advances in Intelligent and Soft Computing", Series, Springer.

[16] Fakhreddine O. Karray and Clarence De Silva.: Soft Computing and Intelligent System Design Theory, Tools and Applications, Pearson Education, 2004.

[17] B.P. Vijay Kumar and P. Venkataram.: Reliable Multicast routing in Mobile networks: a Neural Network approach, IEE proc-communication., vol.150, No.5, pp. 377-384. October 2003.