Optimization of clustering process by hybrid meta-heuristic techniques

Submitted By

Dharmanshu Raval

14MCEN15

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382481

May 2016

Optimization of clustering process by hybrid meta-heuristic techniques

Major Project

Submitted in partial fulfillment of the requirements

for the degree of

Master of Technology in Computer Science and Engineering

Submitted By

Dharmanshu Raval

14MCEN15

Guided By

Prof. Gaurang Raval

Prof. Sharada Valiveti

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382481

May 2016

Certificate

This is to certify that the major project entitled **"Optimization of clustering process by hybrid meta-heuristic techniques"** submitted by **Dharmanshu Raval (Roll No: 14MCEN15)**, towards the partial fulfillment of the requirements for the award of degree of Master of Technology in Computer Science and Engineering of Nirma University, Ahmedabad, is the record of work carried out by him under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project part-I, to the best of my knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

Prof. Gaurang Raval	Prof.Sharada Valiveti
Guide & Associate Professor,	Guide & Associate Professor,
Coordinator M.Tech - NT,	Coordinator M.Tech - INS
Institute of Technology,	Institute of Technology,
Nirma University, Ahmedabad.	Nirma University, Ahmedabad

Dr. Sanjay Garg	Dr P. N. Tekwani
Professor and Head,	Director,
CSE Department,	Institute of Technology,
Institute of Technology,	Nirma University, Ahmedabad
Nirma University, Ahmedabad.	

Statement of Originality

I, **Dharmanshu Raval**, Roll. No. **14MCEN15**, give undertaking that the Major Project entitled "**Optimization of clustering process by hybrid meta-heuristic techniques**" submitted by me, towards the partial fulfillment of the requirements for the degree of Master of Technology in **Computer Science & Engineering** of Institute of Technology, Nirma University, Ahmedabad, contains no material that has been awarded for any degree or diploma in any university or school in any territory to the best of my knowledge. It is the original work carried out by me and I give assurance that no attempt of plagiarism has been made.It contains no material that is previously published or written, except where reference has been made. I understand that in the event of any similarity found subsequently with any published work or any dissertation work elsewhere; it will result in severe disciplinary action.

Signature of Student
Date:
Place:

Endorsed by
Prof Gaurang Raval
(Signature of Guide)

Prof Sharada Valiveti
(Signature of Guide)

Acknowledgements

It gives me immense pleasure in expressing thanks and profound gratitude to **Prof. Gaurang Raval & Prof. Sharada Valiveti**, Associate Professor, Computer Science Department, Institute of Technology, Nirma University, Ahmedabad for his valuable guidance and continual encouragement throughout this work. The appreciation and continual support he has imparted has been a great motivation to me in reaching a higher goal. His guidance has triggered and nourished my intellectual maturity that I will benefit from, for a long time to come.

It gives me an immense pleasure to thank **Dr. Sanjay Garg**, Hon'ble Head of Computer Science and Engineering Department, Institute of Technology, Nirma University, Ahmedabad for his kind support and providing basic infrastructure and healthy research environment.

A special thank you is expressed wholeheartedly to **Dr P. N. Tekwani**, Hon'ble Director, Institute of Technology, Nirma University, Ahmedabad for the unmentionable motivation he has extended throughout course of this work.

I would also thank the Institution, all faculty members of Computer Engineering Department, Nirma University, Ahmedabad for their special attention and suggestions towards the project work.

- Dharmanshu Raval

14mcen15

Abstract

In wireless sensor network deployed in the remote area, because of no rechargeable energy source available, the network lifetime is critically depended on how efficiently we use energy resources. Clustering is the powerful technique to use energy efficiently. Meta-heuristic methods can be applied to do clustering efficiently. So different meta-heuristic methods are analyzed such as PSO, ACO, GA, HS, SA, and its pros and cons are discussed. Also, how those cons are removed in further enhanced version of those methods are given and combinations of these methods are also described with its performance comparison.

Abbreviations

GA	Genetic Algorithm.
LEACH	Low Energy Adaptive Clustering Hierarchy.
PSO	Particle Swarm Optimization
ACO	Ant Colony Optimization
HS	Harmony Search
SA	Simulated Annealing
FCM	Fuzzy C-means

–

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In recent years growth of sensor network is exponential and many new areas are now using sensors. Application of sensors is very crucial in areas like military, health, industry, environmental monitoring and much more. As in many applications, sensors are deployed in remote places, it is not possible to provide energy source or recharge the battery. So the lifetime of the network is mainly determined by how efficiently we use the energy. Clustering is a powerful technique to use the energy efficiently. As it is known that clustering problem is NP-Hard (non-deterministic polynomial time) problem [8]. We have to use some special techniques called meta-heuristic techniques to solve it efficiently. These meta-heuristic techniques are capable enough to get out of local optimum solution and mostly it reaches at the global optimal solution in polynomial time. In this thesis, we have analyzed various techniques like Genetic algorithm, simulated annealing, Particle Swarm optimization, Ant colony Optimization along with conventional clustering algorithms like LEACH and K-means. We have observed that when two of this methods have been combined in one code and then simulated, it gives better results than their individual results, so we have also analyzed the combination of this e.g. PSO-K-means etc.

Chapter 2

Literature Survey

2.1 Conventional Algorithms

2.1.1 LEACH

LEACH [3] is a typical algorithm of clustering which consists of mainly two phase as shown in figure 2.2, one is setup phase and the other is the steady phase. In setup phase, the cluster-heads are selected and cluster setup is done where steps followed are:

1. According to formula given below in equation 1 the nodes which are eligible to become cluster-head advertise themselves in the network.
   Here P is the percentage of the node from total nodes that can be cluster-head, G is a set which contains nodes that are not elected as the cluster head in last r round.

$$T(n) = \begin{cases} \dfrac{P}{1 - P\left(rmod\frac{1}{P}\right)} : n\epsilon G0 : Otherwise \end{cases} \tag{2.1}$$

2. After cluster-head advertisement, the nodes selects their cluster-head based on the shortest distance between them and CH.

3. Then TDMA schedule is defined to make nodes aware when their turn is to communicate with CH.
   CH to node communication is done with TDMA while all CHs are directly connected to Base Station by CDMA scheme.

Flowchart of this algorithm's setup phase is given figure 2.1 given below.[2]

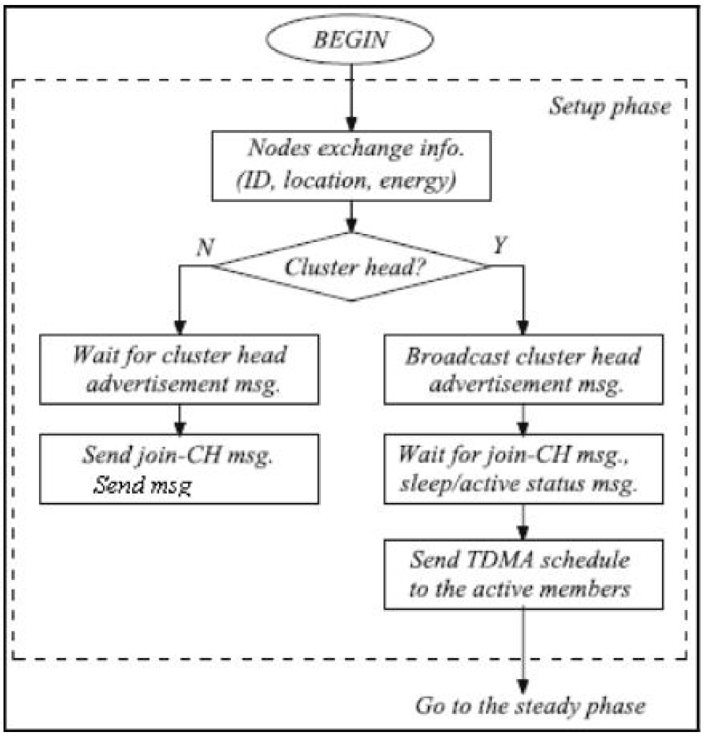

BEGIN
Setup phase
Nodes exchange info. (ID, location, energy)
N
Y
Cluster head?
Wait for cluster head advertisement msg.
Broadcast cluster head advertisement msg.
Send join-CH msg. Send msg
Wait for join-CH msg., sleep/active status msg.
Send TDMA schedule to the active members
Go to the steady phase

Figure 2.1: LEACH[2]

- Problems with LEACH algorithm[9]:

1. CHs are solely defined based on probability and the spatial factor is not present so it is possible to have more CH in close vicinity resulting uneven distribution of CH.
2. If CHs are the nodes at the edge of the network then they have to use more energy and will be drained soon.
3. There is no provision to check residual energy so the node with very low energy can be selected as CH and die soon.

These problems are rectified in further enhanced version of LEACH that is LEACH-C [20]. In that algorithm, the decisions are made at centralized authority which gets data about position and residual energy of every node and decides accordingly. Similarly, P-LEACH that is partition based leach also proposed in which in the first phase the central node computes an optimal number of clusters and partition according to that. Then in the second phase, the CHs are elected for each partition and are advertised so that all other nodes can join their CH according to closest distance [19].

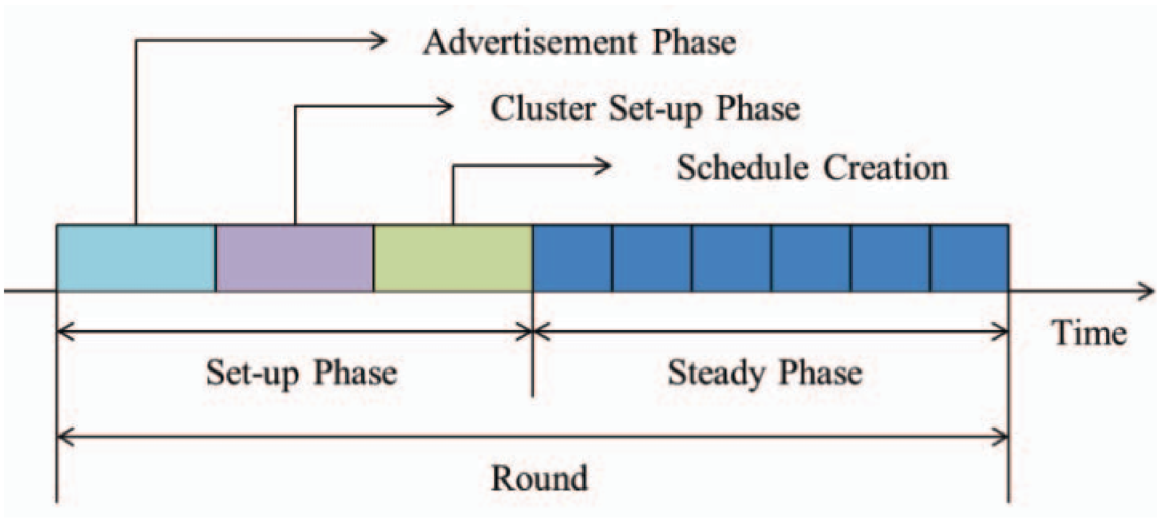


Figure 2.2: LEACH[3]

2.1.2 LEACH-C (LEACH-Centralized)

LEACH protocol has no guarantee for the placement and required number of cluster head nodes in the network. However, employing a centralized algorithm to make the clusters could generate better clusters formation. It has two phases as LEACH: the setup phase and steady state phase. Steady-state phase has been same as LEACH. In the set-up phase of LEACH-Centralized, at the start of each round all nodes send their current energy level and location detail to the base station. To evenly distribute energy load among all the nodes in the network, the base station calculates the average energy of nodes, and those nodes having energy below this average energy cannot become cluster heads for the current round of network. Using the cluster-head nodes, the base station decides clusters using the simulated annealing algorithm[10]. This algorithm makes an attempt to reduce the energy for the non-cluster head nodes to transmit their information to the cluster head, by minimizing the distances between all the non-cluster head nodes and also the nearest cluster head. At every iteration, the next state consists of a collection of nodes in C, is find from the present state, the set of nodes in C, by arbitrarily disturbing the x and y coordinates of the nodes c in C to induce new coordinates x and y. The nodes that have location nearest to (x, y) become the new set of cluster head nodes c that compose set C. k represented by the set of cluster-head nodes C with value f(C) for the current round, the new state, represented by the set of cluster-head nodes C with value f(C), can become the current state with probability

The clustering is done with the following method.

$$f\left(x\right) = \begin{cases} e^{\frac{-\left(f\left(C'\right) - f\left(C\right)\right)}{\alpha_k}} : f\left(C'\right) \geq f\left(C\right) \\ \\ 1 : f\left(C'\right) < f\left(C\right) \end{cases} \tag{2.2}$$

Where the control parameter is α_k and the cost function represents as f( ):

$$\alpha_k = \quad 1000 * e^{k/20} \tag{2.3}$$

$$f\left(C\right) = \sum_{i=1}^{n} min(d^2\left(i, c\right)) \tag{2.4}$$

Where d(i,c) in equation 2.4 represents the euclidean distance between node(i) and node(c). The base station(BS) broadcasts a message with the cluster head ID for each node. If a nodes cluster head ID matches its own ID, the node could be a cluster head; otherwise, the node as non-clusterhead determines its TDMA slot for information transmission and goes to sleep till it's time to transmit information.

2.1.3 K-Means

K-means [4] algorithm mainly consist two steps followed repeatedly. One is the assigning observation to clusters and the other is computing centroid using Euclidean distance to find CH based on current position. This algorithm can be implemented in both centralized and distributed manner. The schematic diagram for the algorithm is as shown in figure 2.2. For two dimensional space and cluster with S nodes the centroid can be found by the equation 2.2 given below.

$$Centroid(X, Y) = \left(\frac{1}{s}\sum_{i=1}^{s} x_i \, \frac{1}{s}\sum_{i=1}^{s} y_i\right) \tag{2.5}$$

The detail steps of algorithm are at [11].

- Centralized vs Distributed approach:

In centralized scheme the information of nodes are transmitted to centralized authority and whatever decision it takes transmitted back to all cluster nodes. The node which is

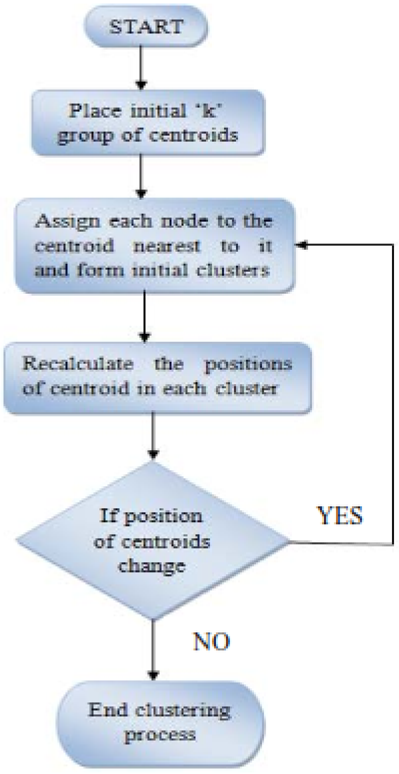

START
Place initial 'k' group of centroids
Assign each node to the centroid nearest to it and form initial clusters
Recalculate the positions of centroid in each cluster
If position of centroids change
YES
NO
End clustering process

Figure 2.3: K-Means[4]

at first distance level from the centroid and highest level of residual energy is selected as CH.

After selecting CH the central authority sends back that information to all the nodes to make aware about to which CH they belong.

While in distributed scheme every node participates in the clustering process. Every node receives information about all other nodes in the network and runs k-means algorithm and decides cluster and CH. Here all nodes having information about their position and their CH the process of transmitting information about the result of clustering process is omitted.

Analyzing the results at [4] we come to know that Distributed scheme makes the network more stable as here task is distributed so the failure of any node make a negligible difference whereas in centralized it make huge if central node is failed.

2.1.4 Fuzzy C-means

FCM algorithm is used for performing centralized clustering. In this algorithm, base station decides cluster heads and cluster members according to the location information of nodes. The aim of cluster formation is to minimize the given objective function [12],

$$J_m = \sum_{i=1}^{N}\sum_{j=1}^{C} \delta_{ij}^{m} \|x_i - c_j\|^2, \quad where \quad 1 \leq m < \infty \tag{2.6}$$

where

$$\delta_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|} \right)^{\frac{2}{m-1}}} \tag{2.7}$$

The term, $\|x_i - c_j\|$ computes the closeness of the data point x_i to the centre vector c_j of cluster j.

After formation of cluster, the base station decides the nearest nodes to cluster centers as a cluster head. Once the cluster formation is finished, BS sends information about the CH and nodes memberships with the cluster. Cluster centers are computed with,

$$C_j = \frac{\sum_{i=1}^{N} \delta_{ij}^{m} . x_i}{\sum_{i=1}^{N} \delta_{ij}^{m}} \tag{2.8}$$

The degree of membership for data point i to cluster j is initialised with a random value θ_{ij} , $0 \leq \theta_{ij} \leq 1$, such that $\sum_{j}^{C} \delta_{ij} = 1$.

The selection process of clusters is repeated in each round with exchanging data among nodes. The CH of each cluster is selected by BS only in the first round. After that next round cluster head will be decided by current cluster heads. Each node attaches its residual energy with the data packet and sends it to the cluster head. According to this energy information, current cluster head will decide new cluster head for next round nearest to the cluster center. Based on the total number of the alive nodes within the cluster, the new cluster head makes a TDMA schedule to allocate the time slot to cluster members. The transmission power of the cluster member is optimized with the help of equation 2.6. Additionally, only cluster members have to turn on their time slot to transmit data, after that they are turned off.

Step by step flowchart is available at 2.4

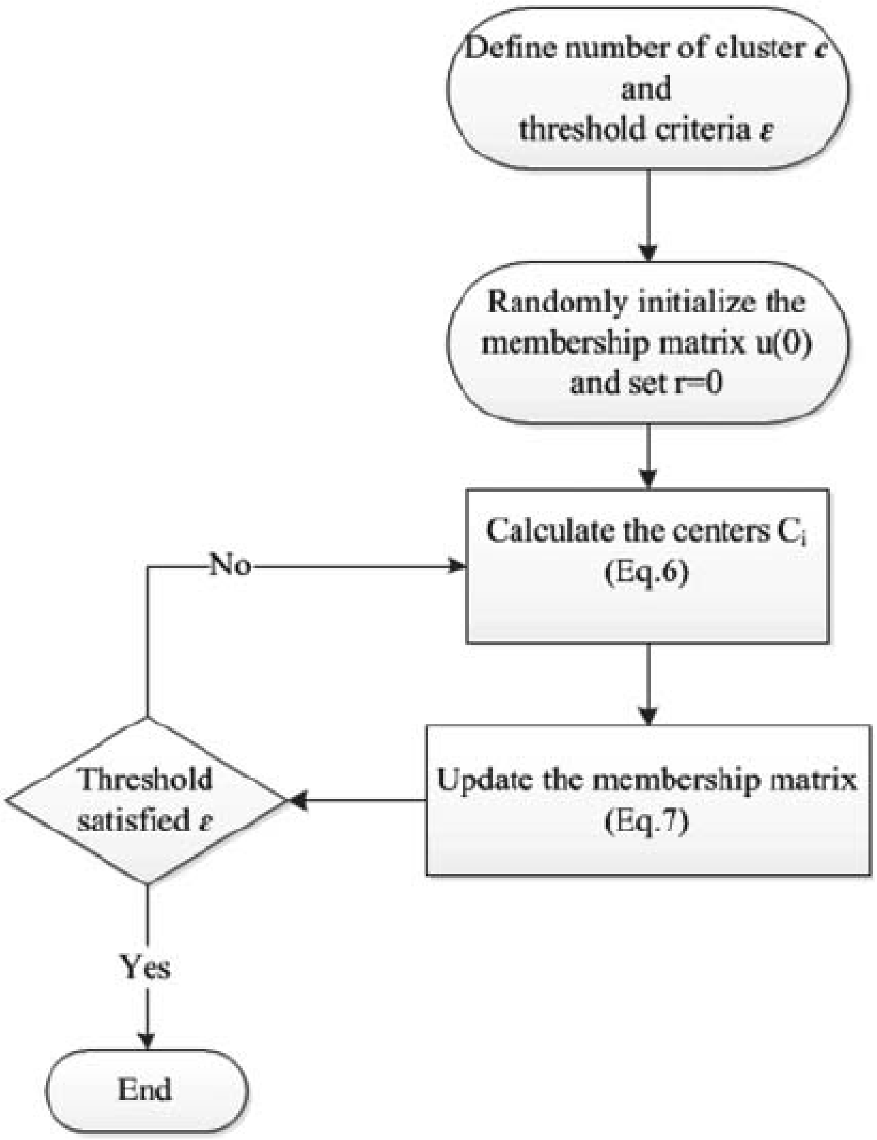

Define number of cluster c
and
threshold criteria ε
Randomly initialize the membership matrix u(0) and set r=0
Calculate the centers C_i (Eq.6)
Update the membership matrix (Eq.7)
Threshold satisfied ε
No
Yes
End

Figure 2.4: Fuzzy C-means[5]

2.2 Optimization With Meta-Heuristic Techniques

Here we analyze clustering with meta-heuristic methods like ACO, HS, GA, SA, PSO and its combinations with some conventional algorithm like K-means.

2.2.1 Optimization with ACO(Ant Colony Optimization)

ACO is one of the ant algorithms which is very powerful in the convergence of NP-hard problems [13].ACO is basically imitating the behavior of real ants which have an incredible ability to find the shortest path from food source to their nest in short time. Ant communicates with the help of pheromone trails which help them to make a decision which path to select [3].

When an ant finds a food source they go to the nest making pheromone trails and it helps other to follow that path. One important thing about pheromone is that it evaporate with time so it prevents from sticking to local optima and helps to converge to global optimum solution [14].

In our WSN clustering perspective ACO works like this [14]:

1. Total N objects are to be covered with K clusters.
2. Main goal is to minimize the Euclidean distance between objects and probable CH
3. So here we use artificial ants to create a solution which is our agent. Every artificial ant makes a probable solution of some size in this iteration using the pheromone level of the previous iteration.
4. As the iteration goes on the trail become more concentrate for the better path and finally from the available trails which are equal to a number of agents in the network, the best trail is chosen and pheromone vector is updated.
5. When the predefined maximum number of iterations are over the process is halted and the latest available solution is near optimal for cluster setup.

The schematic diagram is as given in figure 2.5:

In the simulation results shown by J.Du and L. Wang [14] it is evident that in terms of a number of alive node in comparison with LEACH, LEACH-C and PSO-C this ACO algorithm gives better result[15].

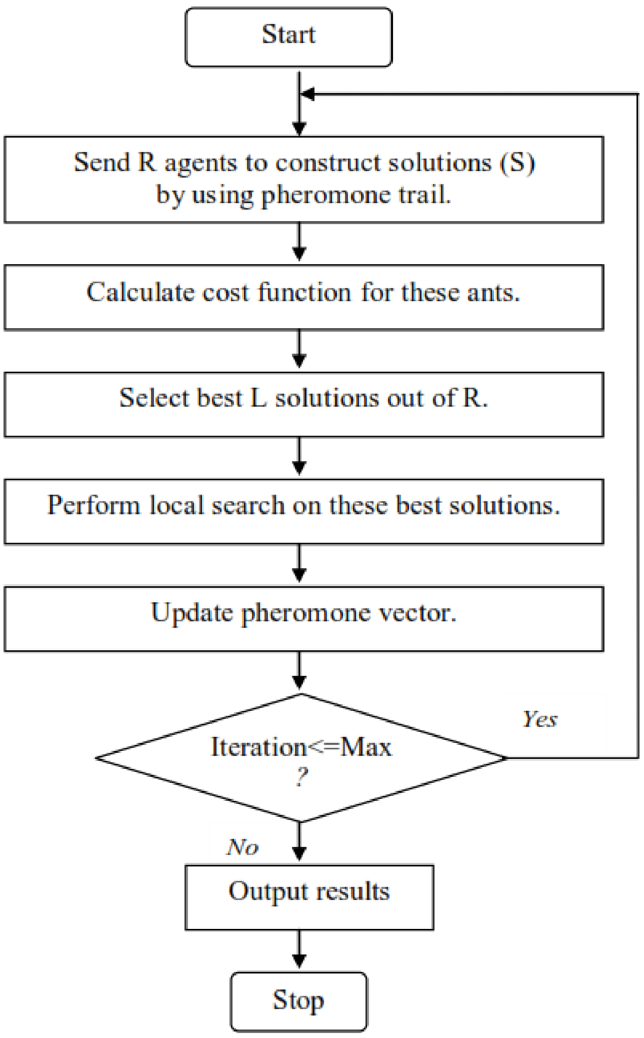

Start
Send R agents to construct solutions (S) by using pheromone trail.
Calculate cost function for these ants.
Select best L solutions out of R.
Perform local search on these best solutions.
Update pheromone vector.
Iteration<=Max ?
Yes
No
Output results
Stop

Figure 2.5: Ant Colony Optimization

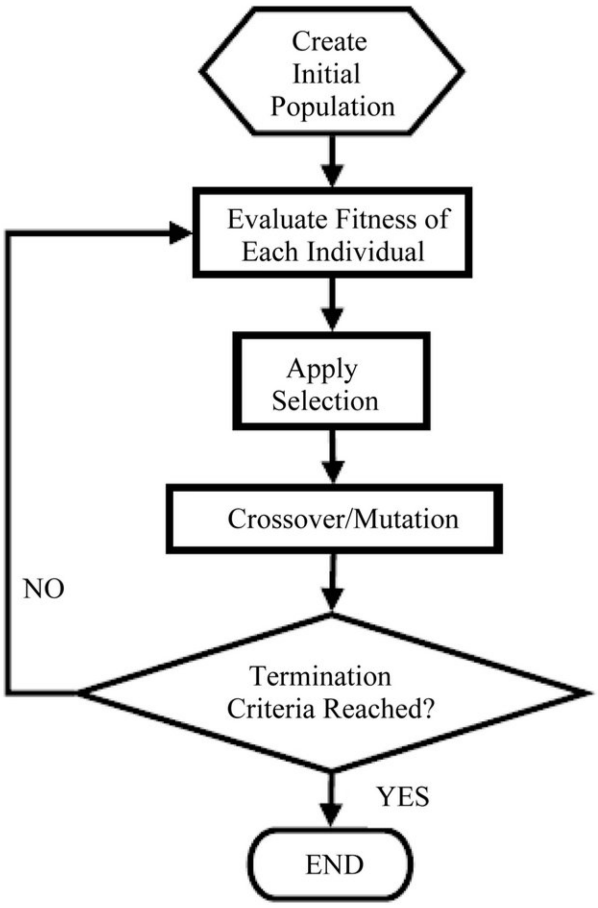

Create Initial Population
Evaluate Fitness of Each Individual
Apply Selection
Crossover/Mutation
NO
Termination Criteria Reached?
YES
END

Figure 2.6: Genetic Algorithm[6]

2.2.2 Optimization with GA (Genetic Algorithms)

Genetic algorithms (GAs) were invented by John Holland in the 1960s, It is basically an adaptive search that mimics natures evolutionary process similar to Darwins survival of fittest. This algorithm makes the random search in the solution space but though randomization, it makes use of past knowledge to take searching in the more converging area in search space. Here search space refers to candidate solutions to a problem and some notion of distance between different candidate solutions [16].

Another important concept in the genetic algorithm is fitness function which defines how much a candidate solution is Fit or optimal in the context of our problem and solution space. As in the natural concept here also chromosome is there which is in terms of candidate solution and all chromosomes are part of randomly generated population.

Flowchart to understand genetic algorithm is as given below in figure 2.6[6]

Important operators in GA:

- Selection: In the process of reproduction this operator selects a particular chromosome if it qualifies fitness criteria. The probability of selection of any particular chromosome varies with how much that chromosome is fit.

- Crossover: This operator imitates biological combination of the chromosome in which two sub-sequences are interchanged with each other from the randomly selected locus to make two offspring.
- Mutation: This operator changes some bits of chromosome string from 1 to 0 and vise-verse.

Algorithm can be explained in some simple steps like this:

1. Randomly generate the population of n different chromosomes.
2. Calculate fitness function for all chromosome.
3. Until n offspring are generated every time select two chromosomes from the solution space with having higher fitness than the threshold and apply cross over to them to form new offspring. Also, apply mutation operator at each locus with mutation rate and put resulting offspring in the new population.
4. Replace the old population with a new one and repeat the procedure from step 2.

In WSN clustering context [11] have used GA to make energy efficient clustering by improving HCR (hierarchical cluster based routing) through GA. Different situation based criteria are considered for making fitness function like (1)Direct distance to sink D (2) Cluster Distance C (3) Cluster Distance Standard deviation SD (4) Transfer energy E (5) Number of Transmission T.

Fitness function for this is as given in equation 2.9:

$$F = \sum \alpha(\omega_i, f_i), \forall f_i \in \{C, D, E, SD, T\} \tag{2.9}$$

Arbitrary weight ω_i is assigned at first but after every iteration it is computed with the equation 2.10.

$$\omega_i = \omega_(i-1) + c_i \triangle f_i \tag{2.10}$$

Where $\triangle f_i = f_i f_(i-1)$ and $c_i = \frac{1}{(1+e^(-f_i - 1))}$

The simulation results at [17] proves that this approach is more energy efficient than traditional clustering algorithms.

2.2.3 Optimization with Harmony Search Algorithm

Harmony search is a meta-heuristic method based on music improvisation concept. As in musical world a musician continuously polishes the pitches to get better harmony, here we search for a better solution from the harmony matrix. Best harmony is analogues to the optimal solution in the context of WSN clustering process.

Pitch adjustment done by musician leads him to get better harmony, in the same way here set of values assigned to each decision variable determine objective function value [18]. In WSN clustering context we can say that musician (decision variable) plays (generates) a note (value) to find better harmony (Global optimal solution).

When a musician is improvising he has three possible option:

1. Play any famous piece of music that already exists. This is similar to get the value directly from harmony memory in our problem that is similar to carry forward good population in the next iteration in GA.

2. Play something similar to known piece with slight variation from available that is pitch adjustment in harmony search and it is similar to cross-over concept in GA.

3. Play an entirely new piece of music created by that musician and that is similar to randomization process in HS.

The flowchart for Harmony search is given in figure 2.7 [7]

Here the first component helps to retain best-searched component for further steps while last increase diversity of the solution that helps to escape from local optimal solution.

Steps in the Harmony Search algorithm are as follows [19]:

1. Initialization of algorithm with various parameter
   Here solution space is defined in terms of harmony matrix with the quantity of HSA (harmony memory size) that is similar to the population in the genetic algorithm.

   Two important parameters are set in this step which is HMCR (harmony memory considering rate) and PAR (pitch adjustment rate). HMCR defines how much values to be considered to carry forward from the past iterations and PAR defines how much modification is allowed in every iteration.Harmony memory is also initialized here by randomly generated solutions at starting if the process.

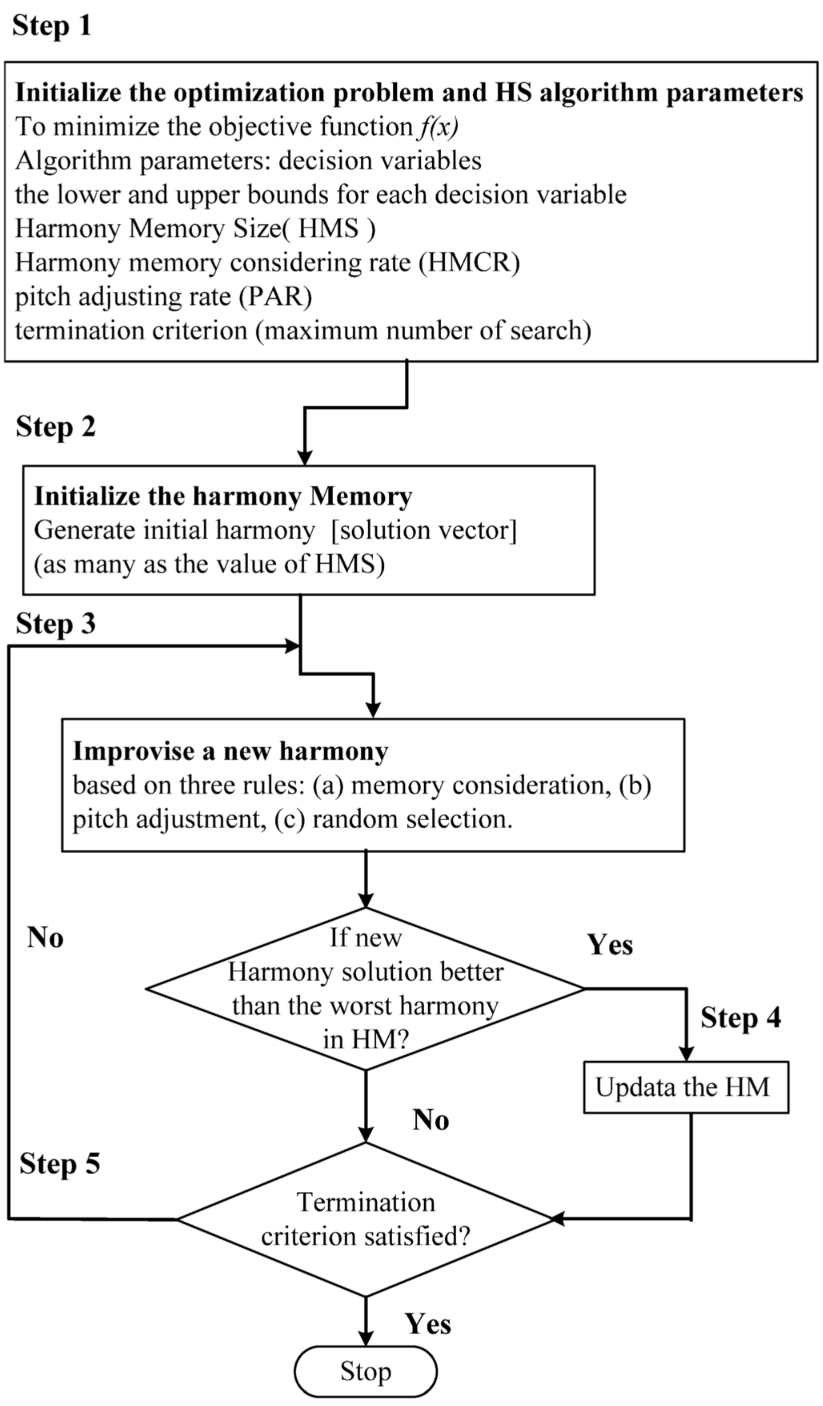

Step 1
Initialize the optimization problem and HS algorithm parameters
To minimize the objective function f(x)
Algorithm parameters: decision variables
the lower and upper bounds for each decision variable
Harmony Memory Size(HMS)
Harmony memory considering rate (HMCR)
pitch adjusting rate (PAR)
termination criterion (maximum number of search)
Step 2
Initialize the harmony Memory
Generate initial harmony [solution vector]
(as many as the value of HMS)
Step 3
Improvise a new harmony
based on three rules: (a) memory consideration, (b) pitch adjustment, (c) random selection.
No
If new Harmony solution better than the worst harmony in HM?
Yes
Step 4
Updata the HM
No
Step 5
Termination criterion satisfied?
Yes
Stop

Figure 2.7: Harmony Search Algorithm[7]

2. Improvise a new harmony from the memory

   Here new harmony vector is generated considering HMCR rate. With the probability of HMCR the next vector will be from HM and with (1-HMCR) the vector will be same as the previous one.

$$I_j^{'} \leftarrow \begin{cases} I_j^{'} \in HM(j) & with\, probability\, HMCR \\ I_j^{'} \in I_{candidate} & with\, probability\, (1 - HMCR) \end{cases} \tag{2.11}$$

   Now if the vector is from HM then it will be further modified with the rate of PAR.

$$I_j^{'} \leftarrow \begin{cases} I_j^{n} \in HM & with\, probability\, PAR \\ I_j^{'} & with\, probability\, (1 - PAR) \end{cases} \tag{2.12}$$

3. Updating Harmony memory

   The new vector generated in the previous step is evaluated by the objective function in this step, if it gives optimized result then it is included in memory otherwise discarded.

   This entire process is repeated until predefined optimization is not reached.

   Simulation results at [20] shows that this new approach gives a better solution than many conventional clustering algorithms like LEACH,K-means, GA, PSO etc.

2.2.4 Optimization with Particle Swarm Optimization

PSO is an imitation of natural qualities of the flock of birds that is having a swarm of s candidate solutions called particles which explore an n-dimensional hyperspace for searching optimal solution [21].

Here two aspects of every particle are important. One is the position of it and the other is the velocity of that particle. An objective function is defined that will evaluate each particle. In the minimization type of optimization the cost of the particle that is near to the optimal solution will be lesser than its father particle and for maximization, it will be higher.

Two measures are defined, one is pbest that is traveling through the solution space the position of that particle when cost is minimum. The other measure is gbest which shows the position of the particle which is best in the entire population.

For every iteration the position and velocity of each particle are updated using the equations in [21].This process is repeated until we find acceptable gbest solution or the maximum number of iteration in done.

Velocity and position are updated with the equation given below:

$$V_i d(k+1) = \omega V_i d(k) + \varphi_1 r_1(k)(pBest_i d - X_i d) + \varphi_2 r_2(k)(gBest_d - X_i d) \tag{2.13}$$

$$X_i d(k+1) = X_i d(k) + V_i d(k+1) \tag{2.14}$$

Where φ_1,φ_2 are constant and $r_1(k)$,$r_2(k)$ are random number uniformly distributed in [0,1] [].
At starting point all particles are given random position and velocity then for each iteration they use their individual best position and global best to maximize the probability of moving towards a better solution space that will result in more fitness then previous iteration [22].

Now in the context of cluster setup the algorithm goes as shown in the figure 2.8:

A new cost function is defined in [22] that takes into account residual energy of to be cluster head node and also the maximum distance from node and CH to minimize the transmission energy and to prevent low energy node from early discharge.

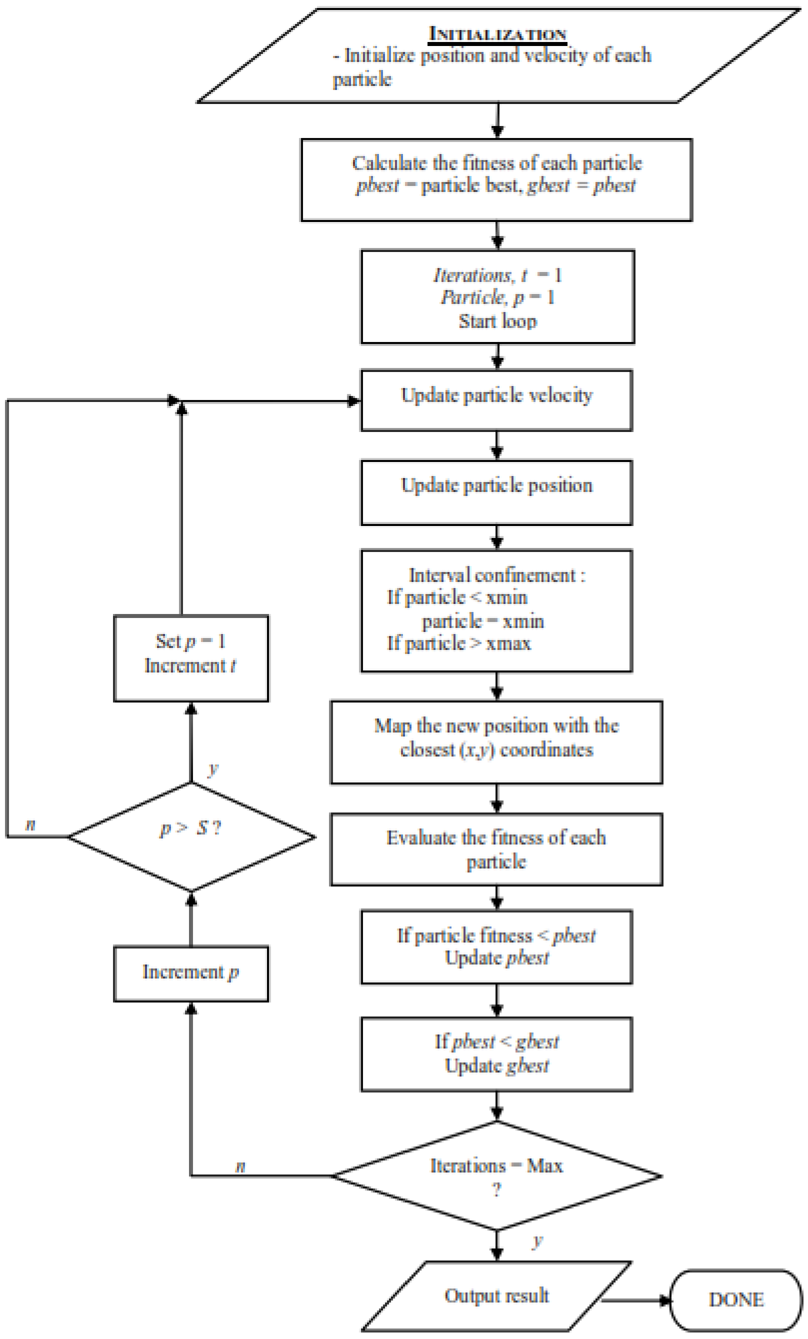


Figure 2.8: Particle Swarm Optimization

2.2.5 Evolutionary Routing Protocol (ERP)

Genetic algorithms are inspired by the theory of Darwin about evolution. A genetic algorithm is started with a set of solutions(described as chromosomes) which is called as population. Solutions are taken from the population and it is used to form a new population. It will be better than the old one. Solutions which are selected according to their fitness value. they are more chances to reproduce new population. Genetic algorithms have large class of evolutionary algorithms (EA), which generate new solutions to optimize problems using natural evolution techniques, such as selection, crossover, and mutation. this process is continued until some condition is satisfied.

An evolutionary algorithm (EA) is a generic and population based on metaheuristic optimization algorithms. An evolutionary algorithm uses method inspired by biological progress, such as selection, recombination, mutation. Generally, EA starts with multiple initial populations of solutions and contains a different type of individual and generate new individual based on variation operators. It has more chance to select the best individual. The value of an individual is calculated based on its fitness.
Each round has two phases: A set-up phase and a steady-state phase.

A set-up phase contains election phase and association phase. In the election phase, CHs nodes are selected using an evolutionary algorithm. The evolutionary algorithm holds a population of a number of chromosome solutions. A complete solution set as a chromosome. It will decide where the CHs (cluster heads) and CMs (cluster members) nodes are located in the wireless sensor network. The set-up phase starts with generating initial population and finds the fitness values for each chromosome in the population. It will be calculated based on the estimated energy utilization, which is determined by the fitness parameters. After the population pass through three evolutionary operators; selection process, recombination and mutation. These operations repeated until the termination criteria are satisfied. Then, select the best the chromosome from the population will be used to start the next phase, where the non CHs nodes are linked to their CHs to form clusters.

In steady-state stage, it is data transmission phase. Several data frames are transmitted from the CMs nodes to their CHs (cluster heads). All the frames are aggregated and then transmitted to the BS (Base Station)[23].

```
Choose an initial population of random solutions;
Evaluate fitness value of each solution;
while the termination criterion is not satisfied do
    Select parents;
    Perform parents recombination;
    Perform offspring mutation;
    Evaluate fitness of the offspring;
end while
```

Algorithm 1: The general framework of an EA

Cluster Head election phase:

1. **Initialize:** Generate an initial EA population. In each chromosome, a non-dead node may be a CH with probability p denoted as 1, otherwise it will be a member node denoted as 0. And dead node is denoted as -1.

$$\forall i \in \{1, ..., n\} \; and \; \forall j \in \{1, ..., \infty, N\} \tag{2.15}$$

$$I_j^i = \begin{cases} 1 & if \; E(node_j) > 0 \; and \; random_j \; \leq \; p \\ 0 & if \; E(node_j) > 0 \; and \; random_j \; > \; p \\ -1 & Otherwise \end{cases} \tag{2.16}$$

2. **Evaluate:** Calculate the fitness value of each individual in the population. The fitness function is to be minimized. The fitness function is given in equation 2.17, where w is a pre-defined value with 0.5. Intra-distance and inter-distance can be computed by equation 2.20 and 2.21 respectively. where CHs defines the total number of clusterheads and C_i is represents i^{th} cluster with CH_i clusterhead. And Compute inter-distance as the minimum Euclidean distance between any two cluster-heads using equation 2.21.

$$Fitness(Chrom_{ERP}) = w \times f_1 + (1 - w) \times f_2 \tag{2.17}$$

$$f_1 = \frac{Compactness}{d_{min}} \tag{2.18}$$

$$f_2 = Number\ of\ clusterheads \tag{2.19}$$

$$Compactness = \sum_{i=1}^{CHs} \sum_{\forall n \in C_i} d(n, CH_i) \tag{2.20}$$

$$d_{min} = \min_{\forall C_i, C_j, C_i \neq C_j} d(CH_i, CH_j) \tag{2.21}$$

3. **Generate:** Generate a new population of chromosomes through:
   (a) **Selections:** Select individual parents from old population set to form the mating pool using binary tournament selection.
   (b) **Recombination:** Create offspring between two consecutive individuals from the mating pool using two point crossover operators with the crossover probability $p_c = 0.6$.
   (c) **Mutation:** Mutate each generated offspring Childs with the mutation probability $p_m = 0.03$.
4. **Replace:** Replace the new population instead of the old one for further process
5. **Evaluate:** Compute the fitness value of each individual in the new population.
6. **Stop:** Repeated until the termination criteria are satisfied.

2.2.6 Energy-aware Evolutionary Routing Protocol (EAERP)

In EAERP[24], Centralized evolutionary algorithm is used to create clusters. Generate an initial population of individuals and each individual is calculated with the help of a fitness function. Then, these individuals will pass through evolutionary operators such as selection, recombination, and mutation with pre-determined crossover and mutate probabilities to increase the quality of the individuals. Each individual consists N nodes, which can be represented 1 for clusterhead, 0 for non-clusterhead, or -1 the for dead node. n individual solutions are specified as in equation 2.15 and 2.16 provided in ERP section. A fitness value of each individual is evaluated using a fitness function. Given objective fitness function is defined as the minimization of the total dissipated energy in the WSN, which is measured as the addition of the total energy dissipated from the non-cluster heads to send data frames to their respective cluster heads, and the total energy dissipated by cluster head nodes to aggregate the data and send the aggregated data to the BS(Base station). The objective fitness function[24] is given in equation 2.22.

$$\varphi_{EAERP}(I^k) = (\sum_{i=1}^{nc}\sum_{s\in c_i} E_{TX_{s,CH_i}} + E_{RX} + E_{DA}) + \sum_{i=1}^{nc} E_{TX_{CH_i,BS}} \tag{2.22}$$

$$E_{TX_{node1,node2}} = \begin{cases} E_{elec} * l + E_{fs} * l * d(node1, node2)^2 & if\ d \leq d_0 \\ E_{elec} * l + E_{mp} * l * d(node1, node2)^4 & if\ d > d_0 \end{cases} \tag{2.23}$$

where nc define the number of clusterheads, s $\in c_i$ is a non-CHs associated with the i^{th} clusterhead node, $E_{TX_{node1,node2}}$ is the energy dissipated for transmitting data frame from node1 to node2. The free space and multipath fading channel models are used to calculate transmitting energy. E_{Rx} is defined as receiving energy as $E_{Rx} = E_{elec} * l$, where E_{elec} is energy dissipated to operate transceiver circuit set to 50 nJ/bit. The data aggregation energy $E_{DA} = 5nJ/bit/message$. The threshold d_0 is calculated by equation 2.24,

$$d_0 = \sqrt{\frac{E_{fs}}{E_{mp}}} \tag{2.24}$$

The next evolutionary operators is the selection operator. It selects best individual from randomly selected two individuals using binary tournament selection from the current population and transfers them for reproduction to the mating pool.

$$I_i^{'} = \begin{cases} I_{i,r1} & if\ \varphi_{EAERP}(I_{i,r1}) \ \leq\ \varphi_{EAERP}(I_{i,r2}) \\ I_{i,r2} & Otherwise \end{cases} \tag{2.25}$$

where, $I_{i,r1}, I_{i,r2}, \forall i \in 1, ..., n$ be two individuals, and r1, r2 ∼ U1, . . . , n are two uniformly distributed random numbers from the set 1, . . . , n. The next evolutionary operators are recombination and mutation. Select two pair of parents from the population for recombination with probability p_c. For each pair, two cut croover point r1, r2 are selected from $1, ..., N-1$. And swap at this two points.

$$R_{\{p_c\}} : I^2 \rightarrow I^2 \tag{2.26}$$

$$I_1^{'} = (I_{1,1}, ..., I_{1,r1}, I_{2,r1+1}, ..., I_{2,r2}, I_{1,r2+1}, ...I_{1,N}) \tag{2.27}$$

$$I_2^{'} = (I_{2,1}, ..., I_{2,r1}, I_{1,r1+1}, ..., I_{1,r2}, I_{2,r2+1}, ...I_{2,N}). \tag{2.28}$$

And each child is mutated with the probability p_m. In mutation, its bit value is inverted from 0 to 1 and vice versa.

$$M_{\{p_m\}} : I \rightarrow I(\forall i \in \{1, ..., n\}\ and\ \forall j \in \{1, ...N\}) : \tag{2.29}$$

$$I_j^{I'} = \begin{cases} I_i^j & if\ I_i^j\ =\ -1\ or\ random\ > p_m \\ 1 - I_i^j & Otherwise \end{cases} \tag{2.30}$$

The *best_I*(best individual) is selected as clustering solution among the population.

$best\_I$ can be specified as:

$$\nexists I \in I^n : \varphi(I) \leq \varphi(best\_I) \tag{2.31}$$

In association phase, The base station decides for all non-cluster head node to which cluster it belongs by using minimum euclidean distances to achieve minimum energy dissipation.

$$\forall j, k \in \{1, ..., nc_{best}\} \quad node_i \in C_j : d(node_i, CH_j) < d(node_i, CH_k) \tag{2.32}$$

2.3 Hybrid Approaches

2.3.1 Optimization with Simulated annealing and Genetic algorithm combined

Annealing is a metallurgical process in which the metal is heated to very high temperature and then gradually the temperature is decreased to give the metal time for getting equilibrium.

As the annealing process is slow, Researchers have determined an annealing schedule that is sufficient for convergence [20].

According to the schedule the temperature we maintain constant temperature for some point of time at each temperature. The annealing schedule can be defined by the equation below:

$$T(t) = \frac{T_0}{log(1+t)} \forall t \geq 0 \tag{2.33}$$

Where $T(t)$ is temperature at time t and T_0 is constant.

As a hybrid approach, AGSAG (adaptive genetic simulated annealing algorithm) is proposed at [20].

A genetic algorithm is not good in searching locally but it has powerful performance overall so it converges slowly whereas simulated annealing is more powerful in searching locally but less aware about overall searching space, so this both algorithms have their own assets and liabilities. If we can combine both than it can complement each other.

In this approach, basic algorithm is same as GA but in the step of mutation SA search for that state which has minimal energy, at the same time GA search for the node having fitness value more than other. So the combined formula at [14] gives more optimal results than its individual algorithms.

2.3.2 K-Means PSO and K-Means GA [1]

Two algorithm namely k-means and particle swarm optimization works good and has their own advantages, but if we can combine both of them to use their positive assets in single algorithm then it will be better than their individual result. KPSO is proposed here to make use of the above-mentioned technique.

In the field of n sensors, the clustering algorithm has to check $2^n - 1$ number of solutions to get the optimal cluster.The proposed approach here is to get the optimal result using two hybrid approaches that are K-means with GA and K-means with PSO.

By partitioning, we decrease the computation effort because now the algorithm will have lesser solution space which is $\frac{n}{k}$

Three phases of this algorithm are like this[1]:

1. Phase 1: Kmeans algorithm is applied to make k number of the partition.
2. Phase 2:PSO/GA algorithm is applied for each partition we got in phase 1
3. Phase 3: We evaluate the resulting cluster layout.

- Phase 1: The k-means clustering algorithm will divide network into K clusters and for each cluster CH will be assigned by the base station, then all non-CH nodes will join the CH nodes.
  Then from the middle of the cluster new CH(at centroid) is selected. The distance between two nodes are computed by the equation below:

  $$D(n_1, n_2) = \sqrt{(xn_1 - xn_2)^2 + (yn_1 - yn_2)^2} \tag{2.34}$$

  Here x and y shows the position of that node by x-coordinate and y-coordinate,this phase divides network into disjoint partitions. Base station saves the information about sensor node ID,its location, and its assigned CH.

- Phase 2:In this phase both GA and PSO algorithms applied individually to the partitions made by the former phase.As an example if 5 partitions are made in phase 1 then this phase will assign 5 CHs,each from one partition. The PSO particle structure and GA chromosome structure are identical, as shown below.

CH_1	CH_2	......	CH_k

  It shows an array that contains an index of each node for every cluster elected as CH. Here, because we have partitioned the network in k clusters the size of this array is known. Fitness function for the PSO and GA algorithm will be as shown below:

$$F = \sum_{i=1}^{allChs} \frac{E_i}{d_i^2} \quad for \quad d_i < d_0 \tag{2.35}$$

$$F = \sum_{i=1}^{allChs} \frac{E_i}{d_i^4} \quad for \quad d_i \geq d_0 \tag{2.36}$$

Here E_i is actual energy of CH and d_i is the euclidean distance between CH and base Station

- Phase 3: In this phase energy consumed by every node is simulated and the equation given below computes Dissipated Energy (DE) for sensing, processing as well as transmission of data.DE_chi is for cluster-head and DE_nonchi is for other nodes[25].

$$DE_{ch_i} = n_i E_e + n_i E_{DA} + \varepsilon_s D_{toBS}^2 for d_i < d_0 \tag{2.37}$$

$$DE_{ch_i} = n_i E_e + n_i E_{DA} + \varepsilon_l D_{toBS}^4 for d_i \geq d_0 \tag{2.38}$$

$$DE_{ch_i} = E_e + \varepsilon_s D_{toCH}^2 for d_i < d_0 \tag{2.39}$$

$$DE_{ch_i} = E_e + \varepsilon_l D_{toCH}^4 for d_i \geq d_0 \tag{2.40}$$

Where

- n_i is the number of nodes belonging to CH_i,
- d_{toBS} is the Euclidean distance between the CH and the base station,
- d_{toCH} is the Euclidean distance between the node and its CH,
- $E_e = 50nj$,$E_da = 5nj$,$\varepsilon_s = 0.0013pj/m^4$,$\varepsilon_l = 0.0013\frac{pj}{m^4}$

2.3.3 KPSO/KGA with Cluster Member Selection

As we know that the life time of network does not just depend on CH but also on each and every node in the network. So in above phase, we have seen the optimization of CH selection, now in this phase we propose the optimization of member selection.

Optimization of member selection is also as important because when the node distribution is un-even and member nodes are connected with the CH that is away from the other CH in nearby then it has to dissipate more energy which can be saved otherwise.

So if we make sure that members will be selected with some predefined scheme then it will again save energy, that we have tried here. The algorithm works here is KPSO-PSO and KGA-GA. It means that the whole process explained in abode section will be done plus one more time the PSO/GA algorithm will work for member selection.

Phase 1: Apply the KPSO/KGA algorithm to partition the network into k Clusters and select the best CH for each cluster.

Phase 2: Cluster member distribution of CHs is obtained by PSO/GAs algorithms considering antenna pattern shape.

Phase 3: Cluster member layout evaluated.

The phase 1 is exactly as it is from the section above.

Phase 2: In this phase we chose optimal cluster members by KPSO/KGA phase using criteria based on antenna pattern shape. The basic concept behind this is that a circular shape is virtually assigned for each CH. Every CH elected by the previous phase this algorithm assign a value r, that is the radius of the virtual shape whose center is cluster-head. Now for that radius if the sensor is covered by that circle than it will be assigned as a member if that CH.

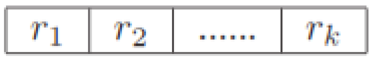
Some rules are specified to make algorithm free from ambiguity.

1. CH which is elected in the previous phase is not allowed to became the member in another cluster even it falls under the virtual circle.

2. When defining r,that is radius it must be less then the euclidean distance of any other CH nearby.

3. If some nodes fall in multiple circles then they will be assigned with nearest CH.

4. If some nodes do not fall in any Circle then it will be assigned to its nearest CH.



r_1	r_2	......	r_k

The gain of the antenna is defined as following equation.

$$G = \frac{S}{A} \tag{2.41}$$

Where S is the area of Isotropic sphere and A is the area of the circular antenna pattern.

$$S = \frac{4D^2}{r^2} \tag{2.42}$$

So for K clusters GA chromosome and PSO particle will be like this:

Where r_1 represent the radius of CH_1.

Fitness Function:

The main objective of fitness function is to make energy consumption less in the network.Here because fitness function has to take in to account many possibilities more terms are considered for minimizing cluster overlappingg.

$$F_2 = E_{dd} + E_{n0}(0.7n_0 + 0.3n_{\geq 2}) \tag{2.43}$$

$$E_{dd} = \sum_{i=1}^{k} x_i \sum_{j=1}^{n_i} (Dist2CH_{ij} + \frac{Dist2BS_i}{n_i}) \tag{2.44}$$

- E_d represents the communication distance,
- k the number of clusters,
- $n_i$is the number of nodes belonging to CH_i
- $Dist2CH_{ij}$ is the square of the Euclidean distance between the sensors and their CH,
- $Dist2BS_i$is the square of the Euclidean distance between the cluster head and the base station,
- n_0 is the number of nodes that do not belong to any CH,

- $n \geq 2$is the number of nodes that belong to more than one CH, and
- En_0 is the sum of the square of the Euclidean distance of all n0nodes to their nearest CH.

Chapter 3

Analysis of Energy Utilization

This chapter covers details of energy usage by the nodes, cluster-heads and by cluster during the rounds.

3.1 Radio Energy Dissipation Model

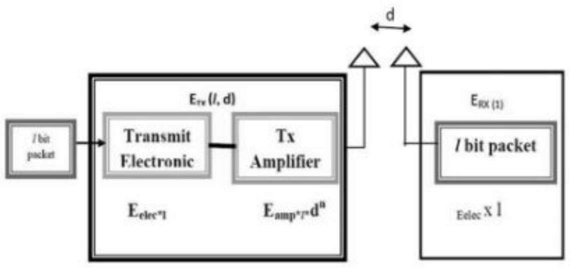


Figure 3.1: Radio model

At transmitter side, energy E_{T_x} is required to transmit L bits at a distance d.

$$E_{TX}(L,d) = \begin{cases} L(E_{elec} + \varepsilon_{fs}d^2) & d < d_0 \\ L(E_{elec} + \varepsilon_{mp}d^4) & d \geq d_0 \end{cases} \tag{3.1}$$

Here d_0 is the deciding factor. It can be use as free space propagation model or multi-path radio propagation model. ε_{fs} and ε_{mp} are the amplification component. It's depend on the propagation model. At receiver side, energy is required to receive L bits,

$$E_{RX}(L) = LE_{elec} \tag{3.2}$$

Energy required for one round,

$$E_{round} = L(2NE_{elec} + NE_{DA} + K\varepsilon_{mp}d^4_{toBS} + N\varepsilon_{fs}d^2_{toCH}) \tag{3.3}$$

Where E_{DA} is energy required to aggregate data, d_{toBS} and d_{toCH} is distance to base station and distance to cluster head respectively and N is number of nodes.

3.2 Energy usage at Setup Phase

Energy required by a Cluster Head during this phase in a round :

$$E_{CH} = k_{opt}E_{Rx} + k_{opt}E_{Tx_{CM}} + k_{opt}E_{Tx_{BS}} + k_{opt}E_{Rx_{CM}} \tag{3.4}$$

where,

$k_{opt}E_{Rx}$ define the energy required by cluster head to receive message of the CHs from base station.

$k_{opt}E_{Tx_{CM}}$ define the energy required by cluster head to transmit the message of the cluster members of that cluster to all its CMs.

$k_{opt}E_{Tx_{BS}}$ define the energy required by cluster head to transmit message of the receipt of base station announcements to base station.

$k_{opt}E_{Rx_{CM}}$ define the energy required by cluster head to receive CMs's join requests.

Energy required by a Cluster Member during this phase in a round :

$$E_{CM} = (N_{nodes} - k_{opt})E_{Rx} + (N_{nodes} - k_{opt})E_{Tx} \tag{3.5}$$

where,

$(N_{nodes}$ - $k_{opt})$ define the total number of CMs within cluster.

$(N_{nodes}$ - $k_{opt})$ E_{Rx} define the energy required by cluster members to receive the cluster head announcements to the CMs.

$(N_{nodes}$ - $k_{opt})$ E_{Tx} define the energy required by cluster members to transmit the join requests to the CHs.

3.3 Energy usage at Steady State Phase

Steady-State phase energy usage is as follows, Energy required by a Cluster Head during this phase in a round :

$$E_{CH} = (\frac{N_{nodes}}{k_{opt}} - 1)lE^{Rx}_{elec} + \frac{N_{nodes}}{k_{opt}}lE_{DA} + lE^{Tx}_{elec} + l\varepsilon_{amp}d^{n}_{toBS} \qquad (3.6)$$

Energy required by a Cluster Member during this phase in a round :

$$E_{CM} = lE^{Tx}_{elec} + l\varepsilon_{amp}d^{n}_{toCH} \qquad (3.7)$$

Energy required by a cluster in a single round :

$$E_{cluster} = E_{CH} + (\frac{N_{nodes}}{k_{opt}} - 1)E_{CM} \qquad (3.8)$$

3.4 Simulation Parameters

We performed a simulation of protocols using ns2 simulator with C++ and tcl. The simulations were performed by varying field size, the number of nodes as mentioned in table 3.1. Base station location was also varied to check the impact of this factor on simulations. The network topologies were created with random node deployments, which are mostly seen in literature. Table 3.2 describe Energy consumption in electronics circuit for transmitting and receiving and data aggregation. The optimal number of cluster heads (K_{opt}) are selected as given in the following equation,

$$K_{opt} = \frac{\sqrt{N}}{\sqrt{2\pi}} \sqrt{\frac{\epsilon_{fs}}{\epsilon_{mp}}} \frac{M}{d^2_{toBS}} \quad [26] \qquad (3.9)$$

where N is total number of nodes and M is the field size.

3.4.1 Topologies with Realistic Deployments

The sensor nodes deployment is easy on a plain surface, open grounds or inside buildings. In these areas, nodes can be kept in a controlled manner to reduce the overall requirement of the resources. But when the area is unknown like mountains or forest area, simple topologies will be difficult to manage while deployment. Different deploying strategies

Parameter	**Value**
Node distribution	(0,0) to (200,200)
BS location	Center, (50,175)
No. of Nodes	50,100,150,200
Initial Node Energy	2J
Simulation Time	3600s
Desired No. of cluster-heads	Optimal as per Equation 3.9
Bandwidth of the channel	1 Mbps
Packet header size	25 Bytes
Message size	500 Bytes

Table 3.1: Simulation Parameter

Operation	**Symbol**	**Energy dissipated**
Energy consumed in electronics circuit for transmitting and receiving	E_{elec}	50nJ/bit
Energy consumed by amplifier to transmit at shorter distance i.e. if $d_{toBS} < d_0$	ϵ_{fs}	$10pJ/bit/m^2$
Energy consumed by Amplifier to transmit at longer distance i.e. if $d_{toBS} \geq d_0$	ϵ_{mp}	$0.0013pJ/bit/m^4$
Energy consumed during data aggregation	E_{DA}	5nJ/bit

Table 3.2: Radio parameters

are required here. In the paper [27], the author has proposed realistic wireless sensor network topologies with a tool called GenSen. Several topologies have been created for 50 sensor nodes to 200 sensor nodes, with BS location varied from center to (50,175) in the field.

Chapter 4

Proposed Approach

4.1 Harmony Search And K-means (Sequential approach)

In our hybrid approach, we use both algorithm, Harmony Search and k-means in such a way that we get the benefit of both algorithms positive sides. Harmony search is very good at the exploration of a large area to find promising areas for the approximate location of CH and K-means algorithm search is good at finding a solution to near proximity with very narrow search span but it gives the precise location in that narrow search space. So here we are using Harmony search algorithm first to find initial approximate CHs and then give that output to k-means algorithm to find more precise CH locations in the second phase. In the third phase, we use Density control approach for assignment of members to already defined CH nodes.

It is also noticed that K-means algorithm starts its execution with random initial means [7]. K-means is very sensitive to this initial means so it makes a huge impact in how the algorithm converges and how much better or near optimal solution we get [9]. Here we have tried to rectify this adverse effect of random means by giving it already optimized means which we get as the output of HS algorithm.

Here we are proposing two varients of this algorithm, One is Sequential hybridization and the other is interleaved hybridization.

Flow chart is given in the figure4.1

This algorithm is shown in following pseudo-code.

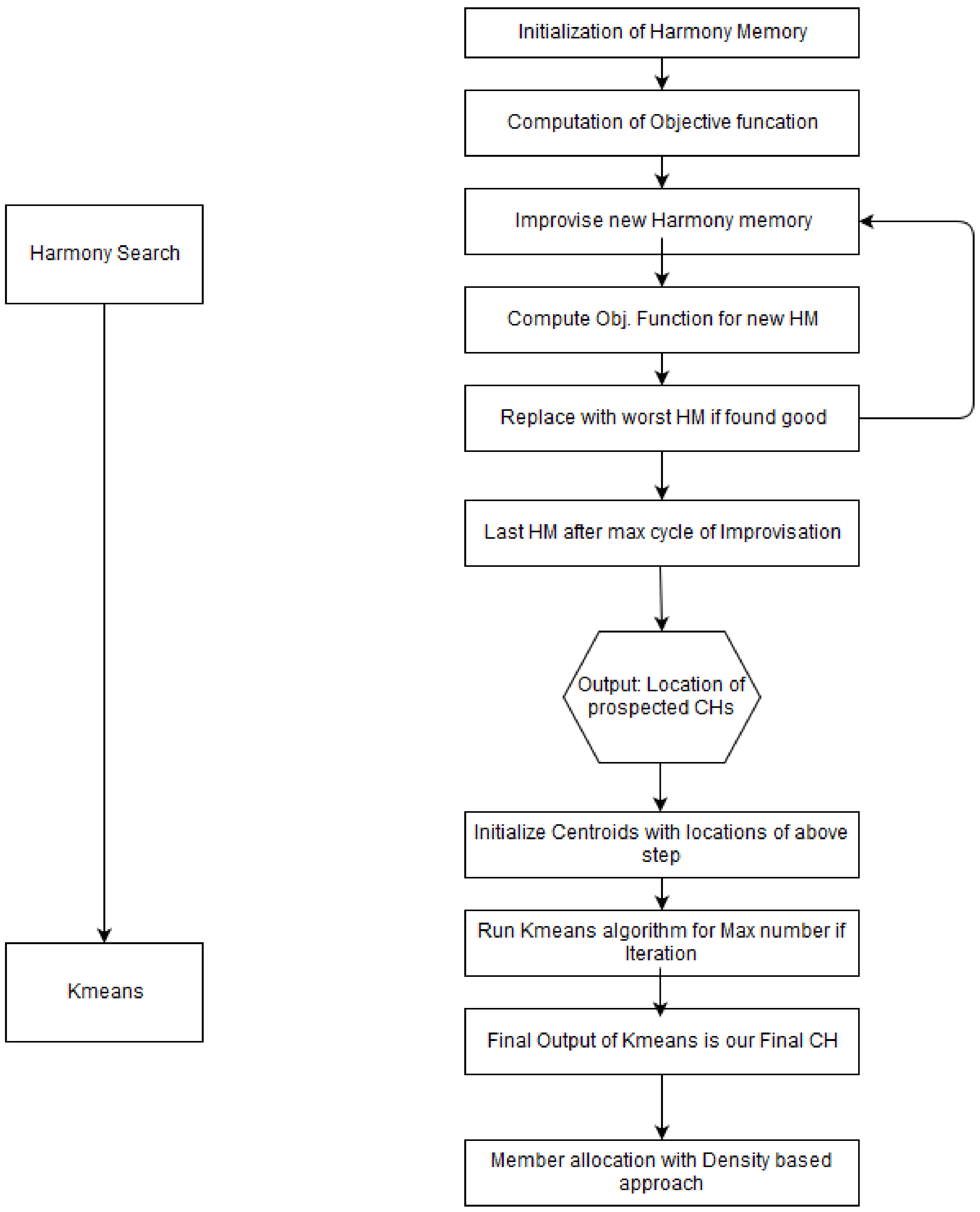

Harmony Search
Kmeans
Initialization of Harmony Memory
Computation of Objective funcation
Improvise new Harmony memory
Compute Obj. Function for new HM
Replace with worst HM if found good
Last HM after max cycle of Improvisation
Output: Location of prospected CHs
Initialize Centroids with locations of above step
Run Kmeans algorithm for Max number if Iteration
Final Output of Kmeans is our Final CH
Member allocation with Density based approach

Figure 4.1: HS-Kmeans (Sequential Hybridization)

Algorithm: Hybridization with HS and K-means (Sequential)

*/*Finding Initial approximate CH*/*

1. Initialize Harmony memory and other parameters like HMCR, PAR etc.
2. Improvise memory using equations 4.2,4.3.
3. Update Harmony memory.
4. Repeat step 2 to 4 for a maximum number of iterations. /*At this step, we get approximate CH as an output of HS algorithm that we give as an input to K-means the algorithm as initial means.*/

   */*refining CH with K-means Algorithm*/*
5. With initial means assign nodes to nearest centroid and form clusters.
6. Recalculate the position of the centroid in each cluster.
7. Repeat 6 and 7 till the centroids continue moving.

   */*At this step, we get the final CH as our output*/*
8. Using Density based approach assign member nodes to the CH we got from the previous step.

Step by Step algorithm is explained as below:

Step 1: *Initialize Harmony memory and other parameters*

Our main goal here is deciding CH in such a way that all the nodes have to spend as less energy as possible so that network lifetime is extended so our target is to minimize function as given in equation 4.4. We store a number of solutions in Harmony Memory that is equivalent to the population of Genetic Algorithm. The format of Harmony memory is given in equation 4.1.

$$HM = \begin{bmatrix} I_1^1 & I_1^2 & ... & I_1^k \\ I_2^1 & I_2^2 & ... & I_2^k \\ ... & ... & ... & ... \\ I_{HMS}^1 & I_{HMS}^2 & ... & I_{HMS}^k \end{bmatrix} \begin{bmatrix} F^1 \\ F^2 \\ ... \\ F^{HMS} \end{bmatrix} \tag{4.1}$$

At the starting point, this memory matrix is initialized with randomly generated solutions and each row of this matrix indicates harmony vector as an optimization solution. For each row, objective function is computed using equation 4.4 and is given by F^J here. Also, the value of HMCR and PAR is initialized as 0.9 and 0.8 in our algorithm.

Here, deciding HMCR properly is very much important because it is the factor that decides how much the available solution in Harmony memory will be considered and how much we will go for randomly generated new solution. With the HMCR rate very high, most of the time next solution vector will be considered from HM. So random searching will be very less. As we know that random exploration is the key factor of meta-heuristic methods to escape from local optimal solution and to reach up to global optimal, very high HMCR will prevent to do that and we will have a less optimal solution. At the same time very low HMCR will always make algorithm go for random exploration and good solutions available in HM will not be considered. So the convergence will be very slow.

Step 2: *Improvisation of New Harmony*

In this step, we improvise a new harmony by generating harmony vector. As defined in the equation 4.2 with the rate equal to HMCR, we consider new vector from Harmony memory and with (1-HMCR) we generate the vector randomly. Now if we have considered new vector from HM then we go for equation 4.3. With the rate equal to PAR we modify or mutate the vector and with (1-PAR) it remains as it is.

$$I_j^{'} \leftarrow \begin{cases} I_j^{'} \in HM(j) & with\, probability\, HMCR \\ I_j^{'} \in I_{candidate} & with\, probability\, (1 - HMCR) \end{cases} \tag{4.2}$$

$$I_j^{'} \leftarrow \begin{cases} I_j^{n} \in HM & with\, probability\, PAR \\ I_j^{'} & with\, probability\, (1 - PAR) \end{cases} \tag{4.3}$$

Step 3: *Update Harmony Memory*

We first evaluate the newly generated vector with the objective function given in the equation 4.4. If the newly generated vector is better than the worst solution available in HM, we replace that worst vector with the newly generated vector.

$$f_{obj_{min}} = \gamma \times f_1 + (1 - \gamma) \times f_2 \tag{4.4}$$

$$f_1 = max_{j\in(1,k)} \left\{ \frac{\sum_{\forall node_i \in C_j} d(node_i, CH_j)}{|C_j|} \right\} \tag{4.5}$$

$$f_2 = \sum_{j=1}^{k} \left\{ \frac{\sum_{\forall node_i \in C_j} V_i^{res}}{V_{CH_j}^{res}} \right\} \tag{4.6}$$

Step 4: *Repetition till maximum iterations*

The above-mentioned steps are repeated up to maximum iterations and after that what we get are the approximate positions of CH. We save these positions and send to the kmeans algorithm as initial means to proceed further.

Step 5: *K-means initialized and execution starts*

We start execution of k-means algorithm taking the output of the previous step as our initial means and other nodes are assigned to that means and clusters are formed.

Step 6: *Movement of means based on assigned nodes*

As we proceed in the k-means, based on the positions of member nodes of every cluster, new means are calculated. With the effect of the new position of new means, member assignment will be done again.

Step 7: *Repetition of above-mentioned procedure*

Till the newly calculated means stops moving from its position, step 5 and 6 are repeated. These means are the final position of Optimal CH and the node situated at that position is assigned as final CH.

Step 8: *Member nodes assignment with Density control*

In the network, it is possible that under some CH so many nodes became members and under some CH, very fewer nodes join as a member, if we advertise CH and allow them to join anyhow. If it happens, the member distribution is uneven and with its effect, energy consumption will also be uneven as denser CH have to spend more energy. Eventually, it can create a hole in the network or it can partition the network and it will affect the overall lifetime of the network. So, we use a systematic way of member assignment with Density control approach. Two types of density control (citation remaining) is given by the equation 4.7. One is strict density control and the other is loose density control.

$$\frac{N}{K} - \delta \leq |C_j| \leq \frac{N}{K} + \delta,\ 1 \leq j \leq K,\ \delta \in (0, K) \tag{4.7}$$

The variation in δ ensures flexibility over the count of members being assigned to the cluster. Due to this variation every cluster size is loosely controlled and also does not vary over a large range. This approach indirectly improves energy efficiency also as it does not force far away nodes to become the member of the cluster. It also brings in uniform data delivery from sub-regions of the deployment field.

4.2 Harmony Search And K-means (Interleaved approach)

In this variant, I have tried to make the process more hybrid and more dependent on each-other. Flow chart is given in the figure 4.2

This algorithm is shown in following pseudo-code.

Algorithm: Hybridization with HS and K-means (Interleaved)

```
/*Finding Initial approximate CH*/

1. Initialize Harmony memory and other parameters like HMCR, PAR etc.
2. Improvise memory using equations 4.2,4.3.
3. Update Harmony memory.
4. Repeat step 2 to 4 for a maximum number of iterations. /*At this step, we get
   approximate CH as an output of HS algorithm that we give as an input to K-means
   the algorithm as initial means.*/
   /*refining CH with K-means Algorithm*/
5. With initial means assign nodes to nearest centroid and form clusters.
6. Recalculate the position of the centroid in each cluster.
7. Repeat 6 and 7 till the centroids continue moving.
8. Evaluate Final Output of k-means with respect to Objective Function of HS algorithm.
9. Replace the worst HM with the new solution if found better than worst available
   in HM.
```

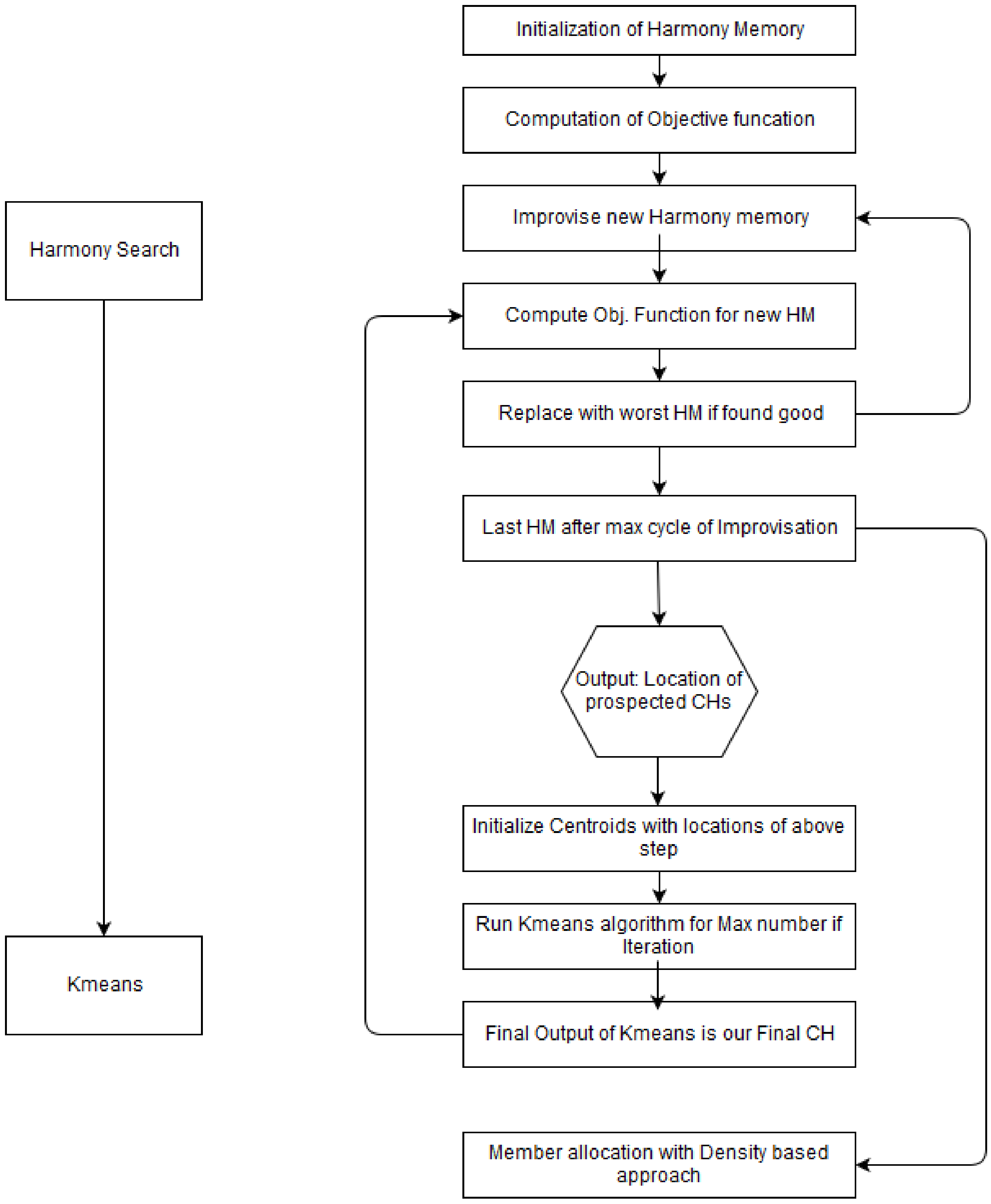

Initialization of Harmony Memory
Computation of Objective funcation
Improvise new Harmony memory
Compute Obj. Function for new HM
Replace with worst HM if found good
Last HM after max cycle of Improvisation
Output: Location of prospected CHs
Initialize Centroids with locations of above step
Run Kmeans algorithm for Max number if Iteration
Final Output of Kmeans is our Final CH
Member allocation with Density based approach
Harmony Search
Kmeans

Figure 4.2: HS-Kmeans (Interleaved Hybridization)

10. Repeat from step 2 for a maximum number of cycles.

11. Using Density based approach assign member nodes to the CH we got from the previous step.

In this algorithm, the final output is taken from the last harmony memory while in sequential approach the final output is considered from the output of K-means algorithm. Here because we are evaluating the output of K-means algorithm with respect to the objective function of Harmony Search, after every cycle is completed we have two potential solutions available, One from Improvisation of Harmony memory itself and the other is from K-means.

Because of this reason the interleaved approach gives more optimized result than the sequential approach.

4.3 Fuzzy C-means and Harmony Search

In this approach, two algorithms namely Fuzzy C-means and Harmony search are combined to have a hybrid approach. First, the FCM algorithm is running for its maximum iterations and the output of this algorithm in terms of prospective CH locations are given to Harmony Search algorithm at the time of Initialization of Harmony Memory. In harmony Memory, this output is considered as a potential solution along with other solutions those are generated randomly.

The Flowchart given in figure 4.3 gives a better idea about detail steps and these steps are explained in the further text.

This algorithm is shown in following pseudo-code.

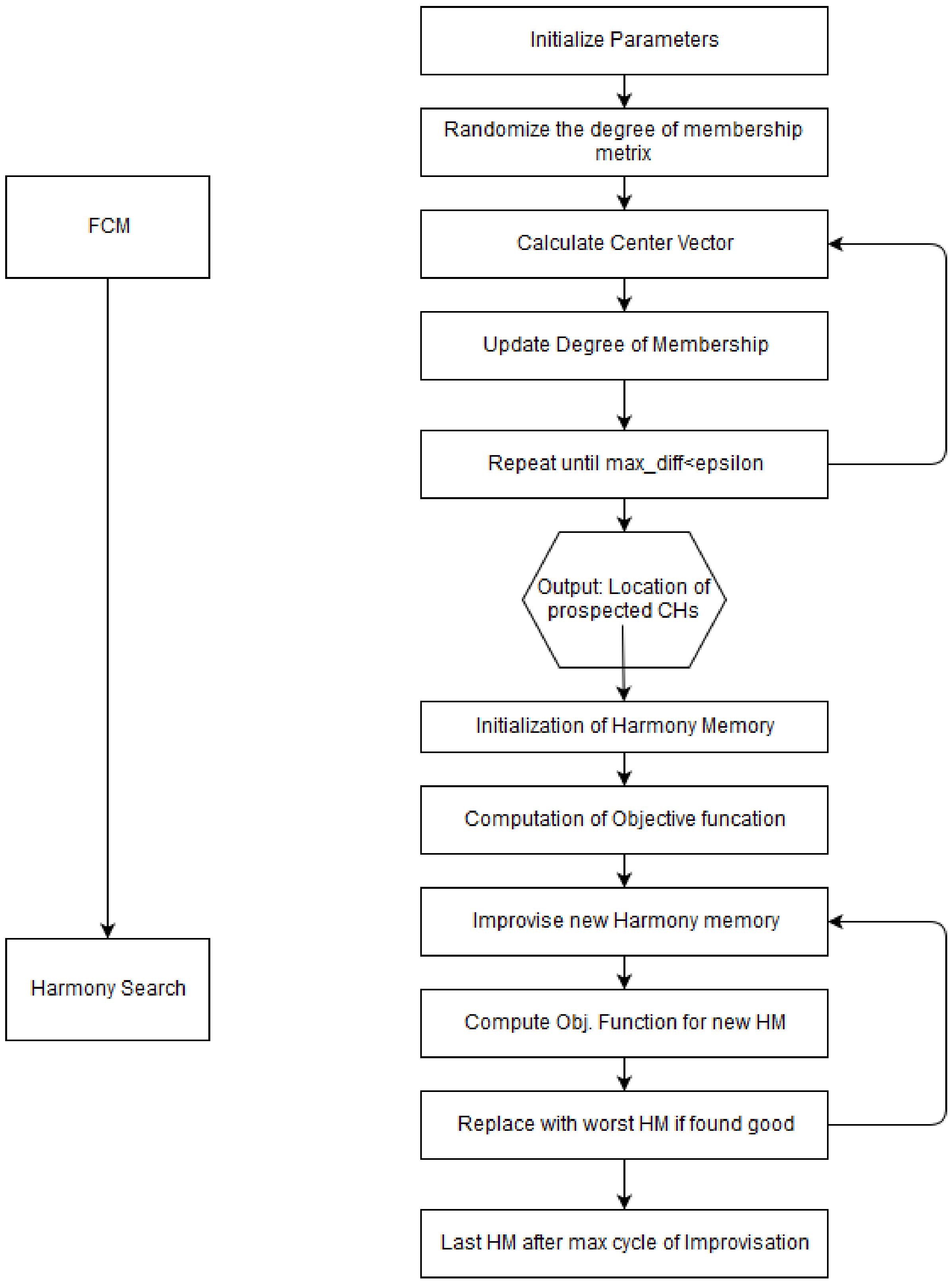
Algorithm: Hybridization with HS and K-means (Interleaved)

*/*Finding Initial approximate CH*/*

1. Initialize various parameters like fuzzy Co-efficient, the degree of membership etc. for the FCM algorithm.

2. Randomize the Degree of membership matrix.

3. Calculate Cluster center with the equation 4.8


FCM
Harmony Search
Initialize Parameters
Randomize the degree of membership metrix
Calculate Center Vector
Update Degree of Membership
Repeat until max_diff<epsilon
Output: Location of prospected CHs
Initialization of Harmony Memory
Computation of Objective funcation
Improvise new Harmony memory
Compute Obj. Function for new HM
Replace with worst HM if found good
Last HM after max cycle of Improvisation
Figure 4.3: FCM and Harmony Search Hybrid approach

4. Update Degree of membership using the equation 4.9

5. Repeat above steps until termination criteria.

   /* At this step, we have prospected CH locations and Node IDs from the output of FCM algorithm those are to be given to HMS in the next step.

6. Initialize Harmony memory and other parameters like HMCR, PAR etc.

7. Improvise memory using equations 4.2,4.3.

8. Update Harmony memory.

9. Repeat step 2 to 4 for a maximum number of iterations.

10. Final output of HS is considered for the formation of Clusters.

Step by Step Algorithm is explained below:

Step 1: *Initialization of Parameters for FCM*

In this step, various parameters of FCM algorithm are initialized like data points which refer to sensor nodes in our case, the degree of membership,fuzziness co-efficient etc.

Step 2: *Randomize Degree of Membership*

In this step for the purpose of random exploration, which is very much necessary for reaching up to the global optimal solutions, the degree of membership for each node is randomly generated. Here as we know that we are not assigning the node to any one particular cluster but we compute the probability of having this node in every cluster and that is represented by the degree of membership, computing this parameter is very much crucial for successfully running the algorithm.

Step 3: *Calculate Center vectors based on degree of membership*

In this step, based on the available degree of membership we compute cluster centers using the equation 4.8

$$C_j = \frac{\sum_{i=1}^{N} \delta_{ij}^{m} . x_i}{\sum_{i=1}^{N} \delta_{ij}^{m}} \tag{4.8}$$

The degree of membership for data point i to cluster j is initialised with a random value θ_{ij} , $0 \leq \theta_{ij} \leq 1$, such that $\sum_{j}^{C} \delta_{ij} = 1$.

Step 4 *Update Degree of Membership* In this step we update degree of membership based on computed cluster center using the equation 4.9

$$\delta_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|} \right)^{\frac{2}{m-1}}} \tag{4.9}$$

The term, $\|x_i - c_j\|$ computes the closeness of the data point x_i to the centre vector c_j of cluster j.

Basically we want to minimize the Objective function given at 4.10

$$J_m = \sum_{i=1}^{N} \sum_{j=1}^{C} \delta_{ij}^{m} \|x_i - c_j\|^2, \quad where \quad 1 \leq m < \infty \tag{4.10}$$

Step 5: *Repeat above steps until termination criteria*

The criteria for termination ϵ is defined from starting point and we compare the value of J_m with that and until $J_m < \epsilon$ the steps are repeated. After completion of this algorithm, the output is available in terms of Node IDs and Node locations of prospected Cluster Heads and those are given to the Harmony search at the time of Initialization.

Step 6: *Initialization of HS Parameters*

In this step various parameters,including Harmony memory, are initialized but the difference here from the other algorithm is, here total (HM size-1) number of solutions are randomly generated and one solution is directly taken from the output of FCM algorithm.

Step 7: *Improvisation of HM*

In this step as in normal HS is done, Improvisation of Harmony memory is completed with equations 4.2,4.3.

Step 8: *Update Harmony memory*

After evaluation of newly generated Harmony, if found better than worst available in HMS, the newly generated solution is replaced with the worst solution.

Step 9: The above-mentioned process is repeated until the maximum number of cycles are completed and after that final output is considered for the formation of Clusters.

Chapter 5

Simulation Results

We have simulated the proposed algorithms for four different topologies namely, 50nodes, 100nodes, 150nodes and 200nodes. The base station is located at center of the deploying Field that means for 50 nodes topology the base station is located at (25,25) and like vise.

In the graphs given below we have merged all the algorithms for two type of result values namely data transfer and alive nodes for every topology.

5.1 Data transfer and Alive nodes Results

5.1.1 50nodes Simulation Results

In figure 5.1 We can see the data transfer of each algorithm namely HS-Kmeans(Seq), Kmeans, Harmony search, Leach-C, FCM, HS-Kmeans(Int) and FCM-HS hybrid method. Here three algorithms, HS-Kmeans(Seq),HS-Kmeans(Int),FCM-HS are our proposed approach and others are individual meta-heuristic methods. We can see that FCM-HS have highest performance in this graph and other two proposed are also giving better result than their individual algorithms.

In figure 5.2 we can see the performance in terms of aliveness of nodes for different algorithms. Here we can see that FCM-HS is giving better stability because it has more intermediate level sustainability than other algorithms. We can also see that in terms of last node die performance HS-Kmeans(seq) is giving better results,though it has lower stability in intermediate stages but at last it is having alive nodes still there when all the other algorithms are not having single ailve node available.

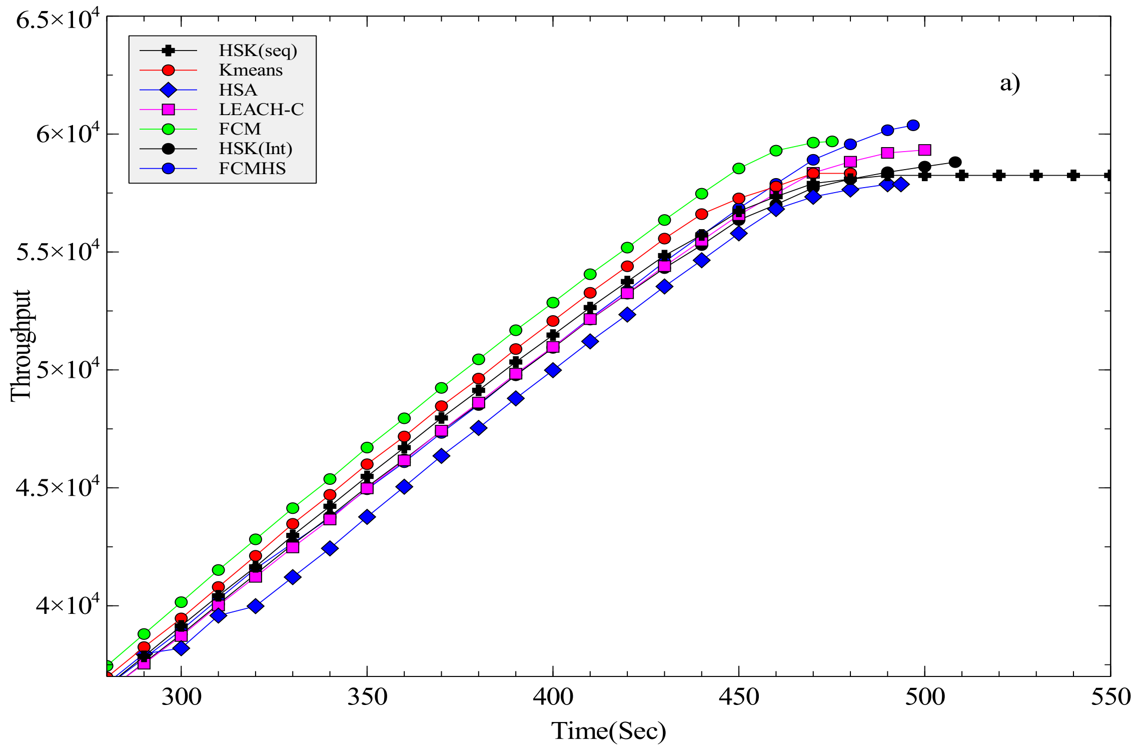


Figure 5.1: Per Node Data Delivery - 50 Nodes

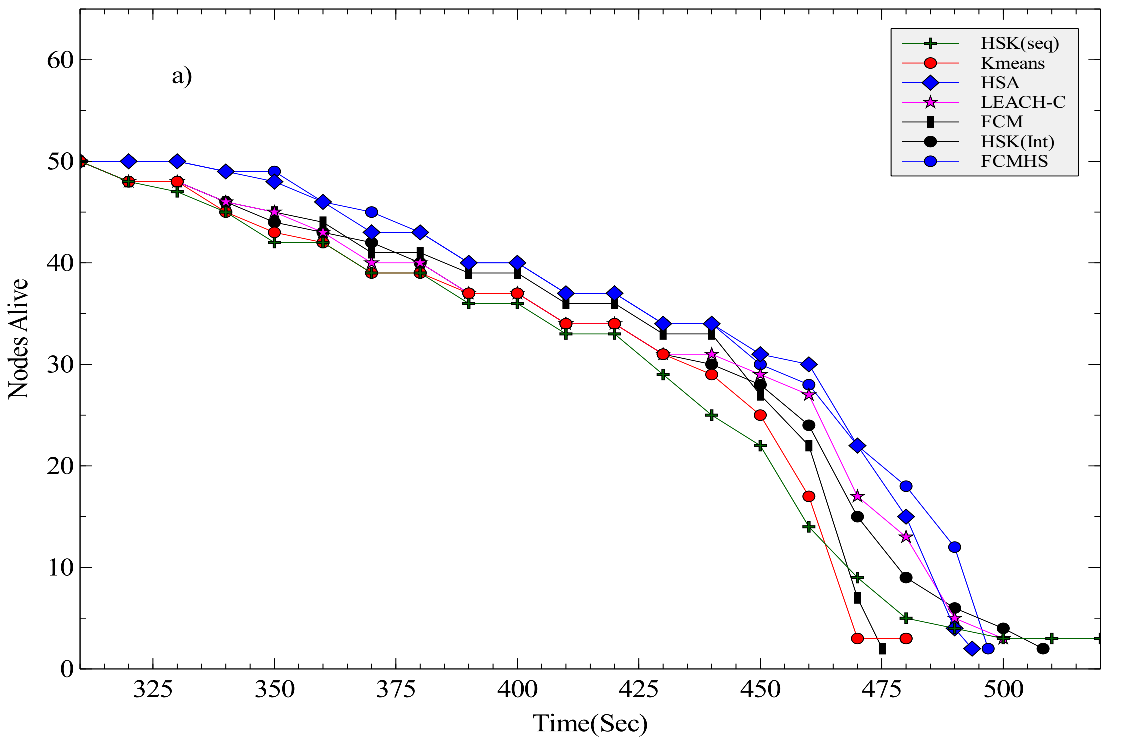


Figure 5.2: Alive nodes - 50 Nodes

5.1.2 100nodes Simulation Results

In the figure 5.3 We have graph for data delivered for all algorithm and we can see that here all algorithms are almost giving similar performance but the proposed approaches have slightly better performance.

While in terms of aliveness if we see in 5.4 for 100nodes topology, we can see that the HS-Kmeans(Seq) approach is giving better result if we consider last node die scenario.

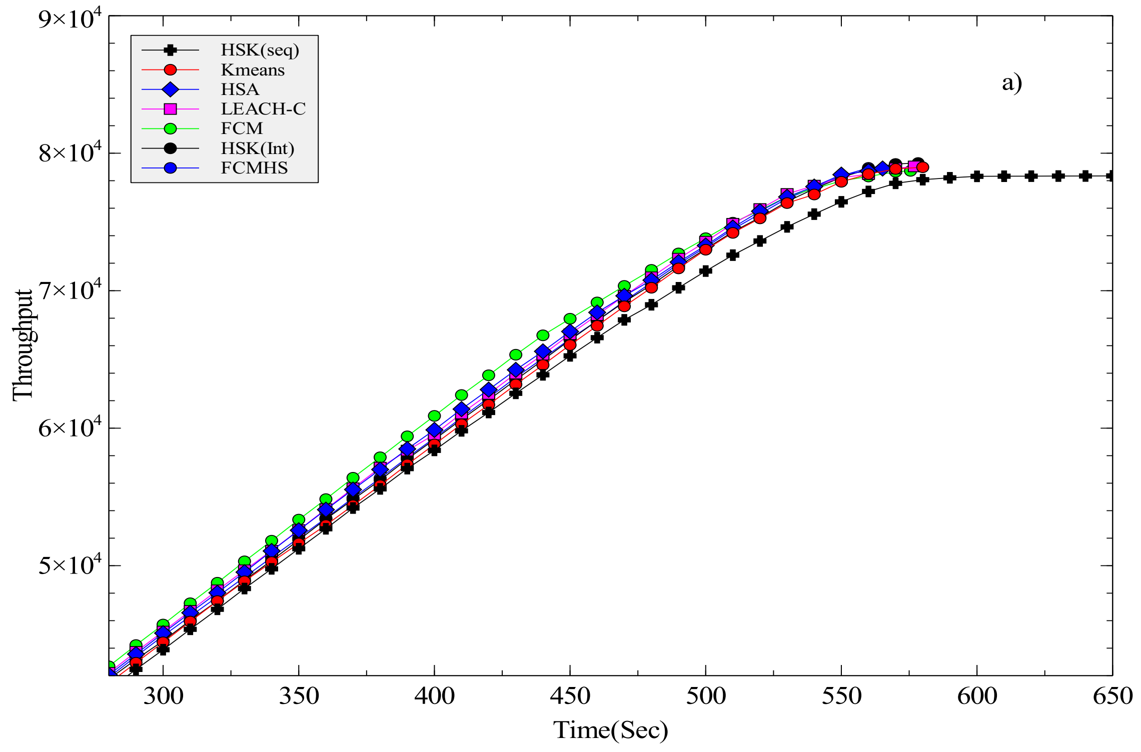


Figure 5.3: Per Node Data Delivery - 100 Nodes

5.1.3 150nodes Simulation Results

In figure 5.5 we can see the clear difference between the graphs of proposed approach and other individual methods, here FCM-HS is giving best result out of all algorithms and HS-Kmeans(Int) method also gives very good result. HS-kmeans(Seq) have moderate result available but it is better than the individual methods also.

In figure 5.6 we can see the aliveness graph that shows that in terms of aliveness the HS-Kmeans(Seq) gives far more better results than other and we can also see that the FCMHS approach have higher stability that can be seen by first node die time.

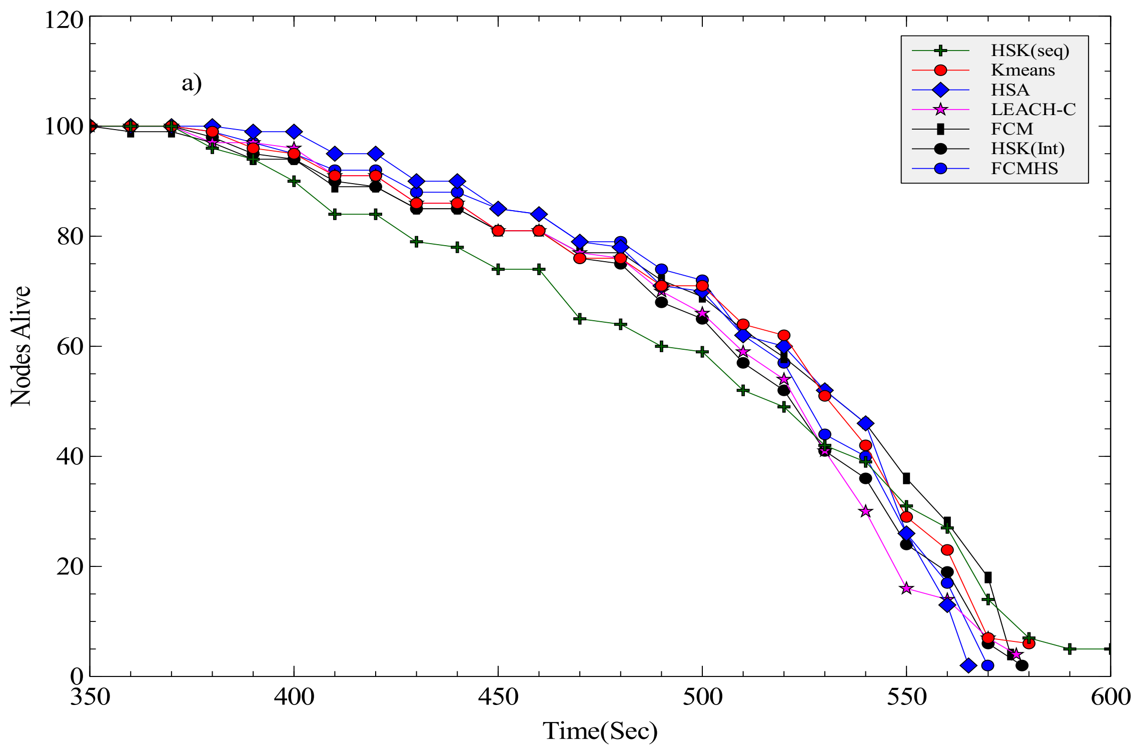


Figure 5.4: Alive nodes - 100 Nodes

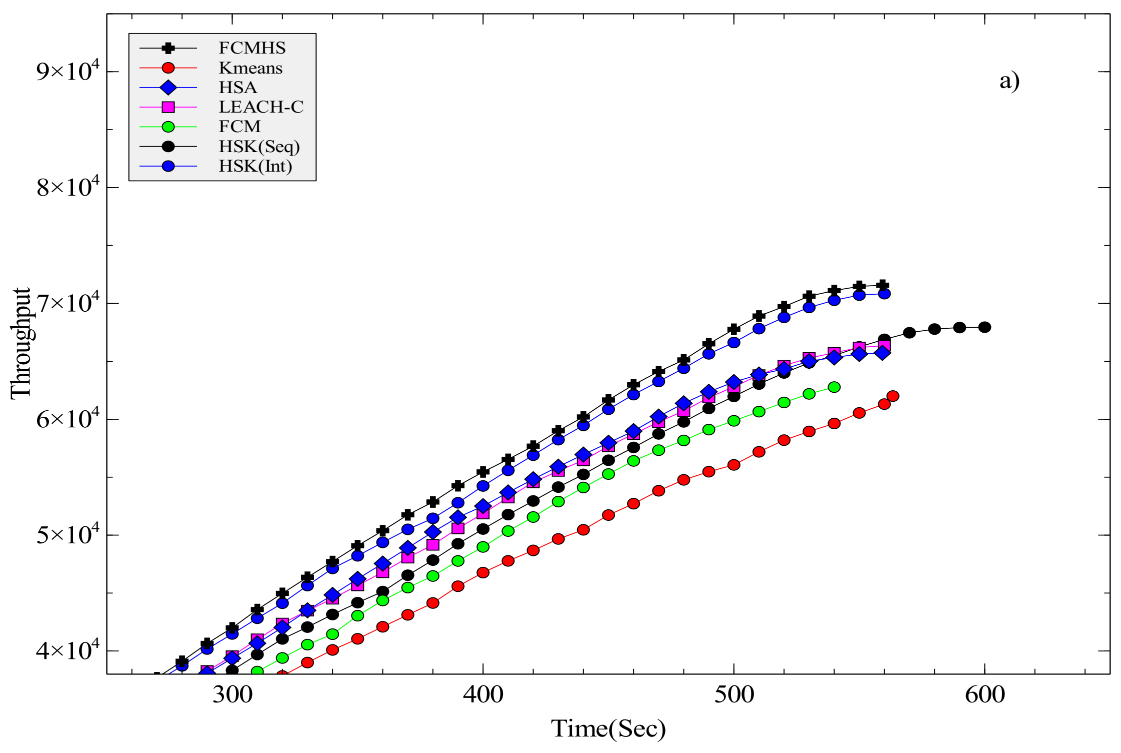


Figure 5.5: Per Node Data Delivery - 150 Nodes

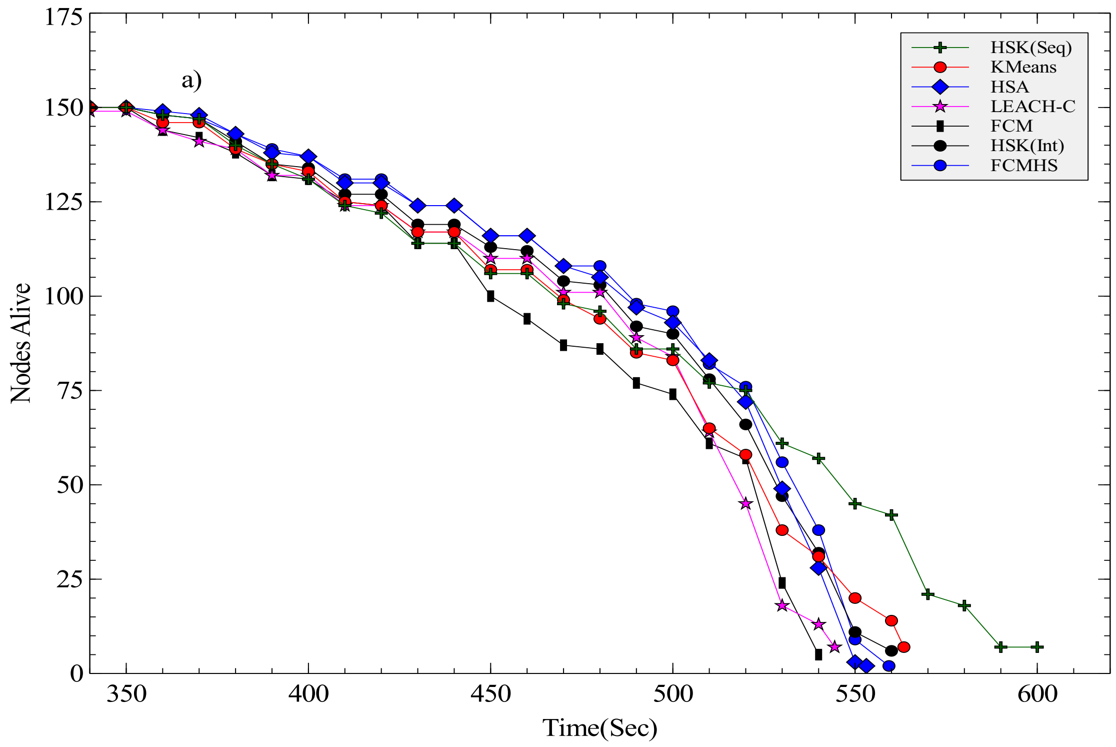


Figure 5.6: Alive nodes - 150 Nodes

5.1.4 200nodes1 Simulation Results

In figure 5.7 for data transmition related graph we can see that FCMHS is at the end giving more better results whlie HS-Kmeans(seq) is not giving appropriate result may be because it is facing stability problem when number of nodes increases after some limit. But HS-Kmeans(Int) is not facing similar difficulty because of different hybridization mechanism.

In figure 5.8 for alive nodes result we can see that again HS-Kmeans(seq) is giving better result than all other algorithms combined, and we can also see that FCMHS is providing very good stable results from starting point although in terms of Last node die, the HS-Kmeans(seq) is performing better.

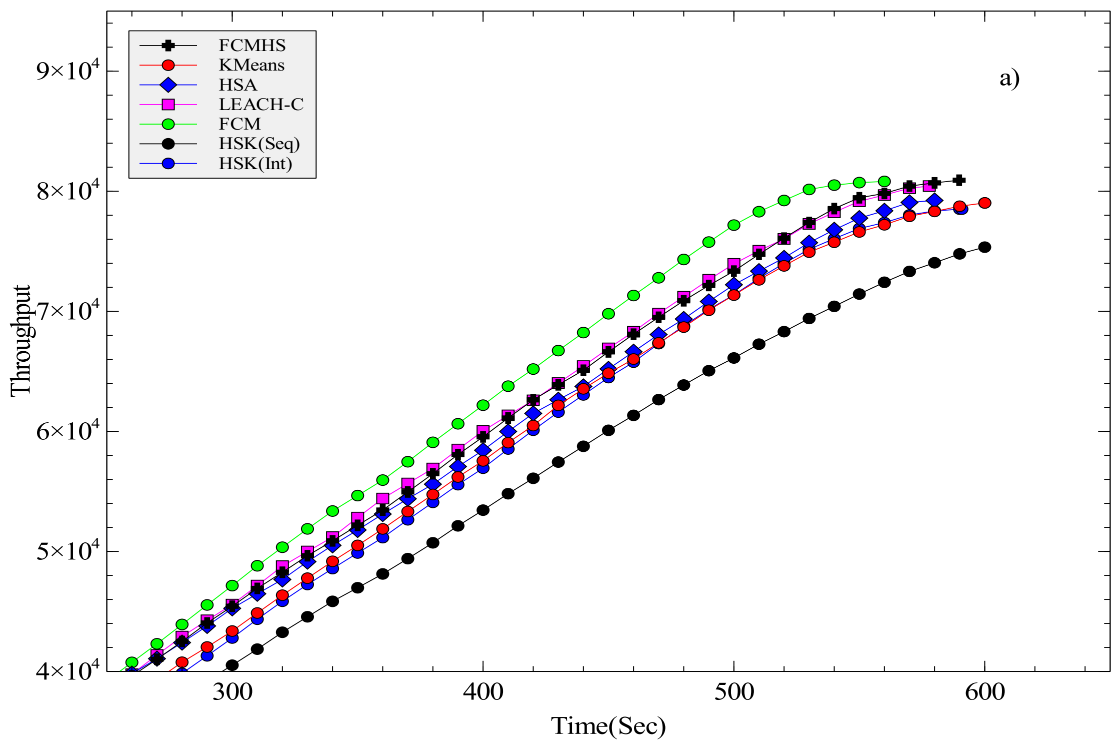


Figure 5.7: Per Node Data Delivery - 200 Nodes

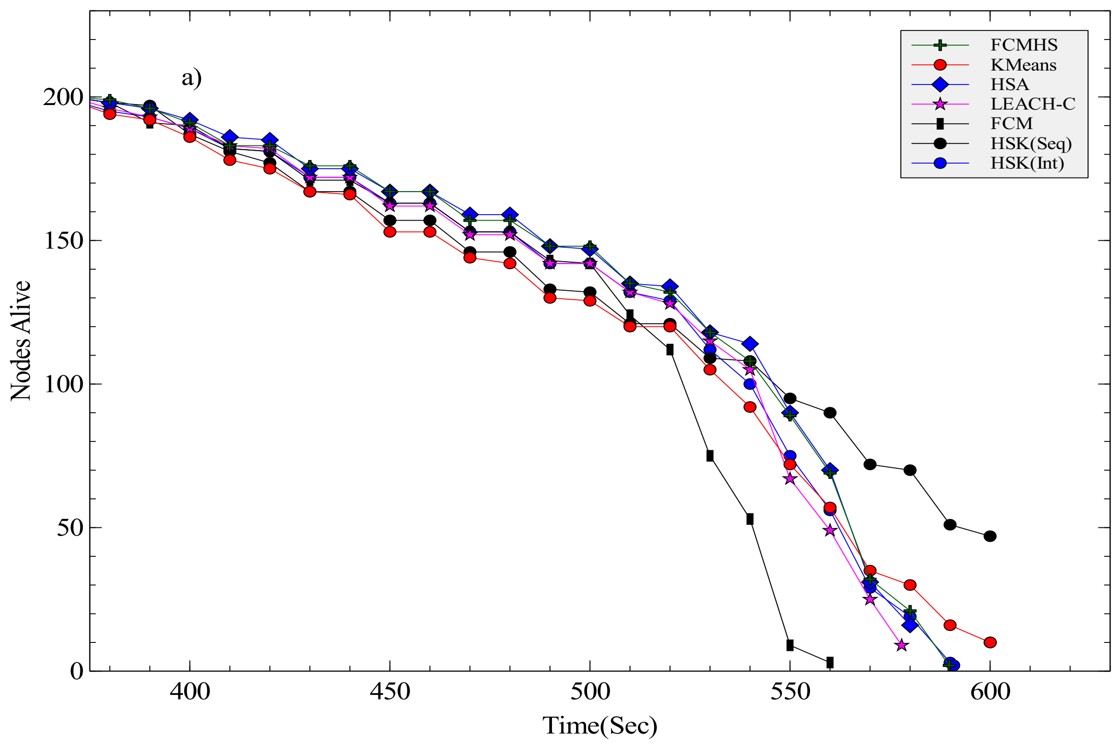


Figure 5.8: Alive nodes - 200 Nodes

5.1.5 Bar chart of Alive nodes for all topology

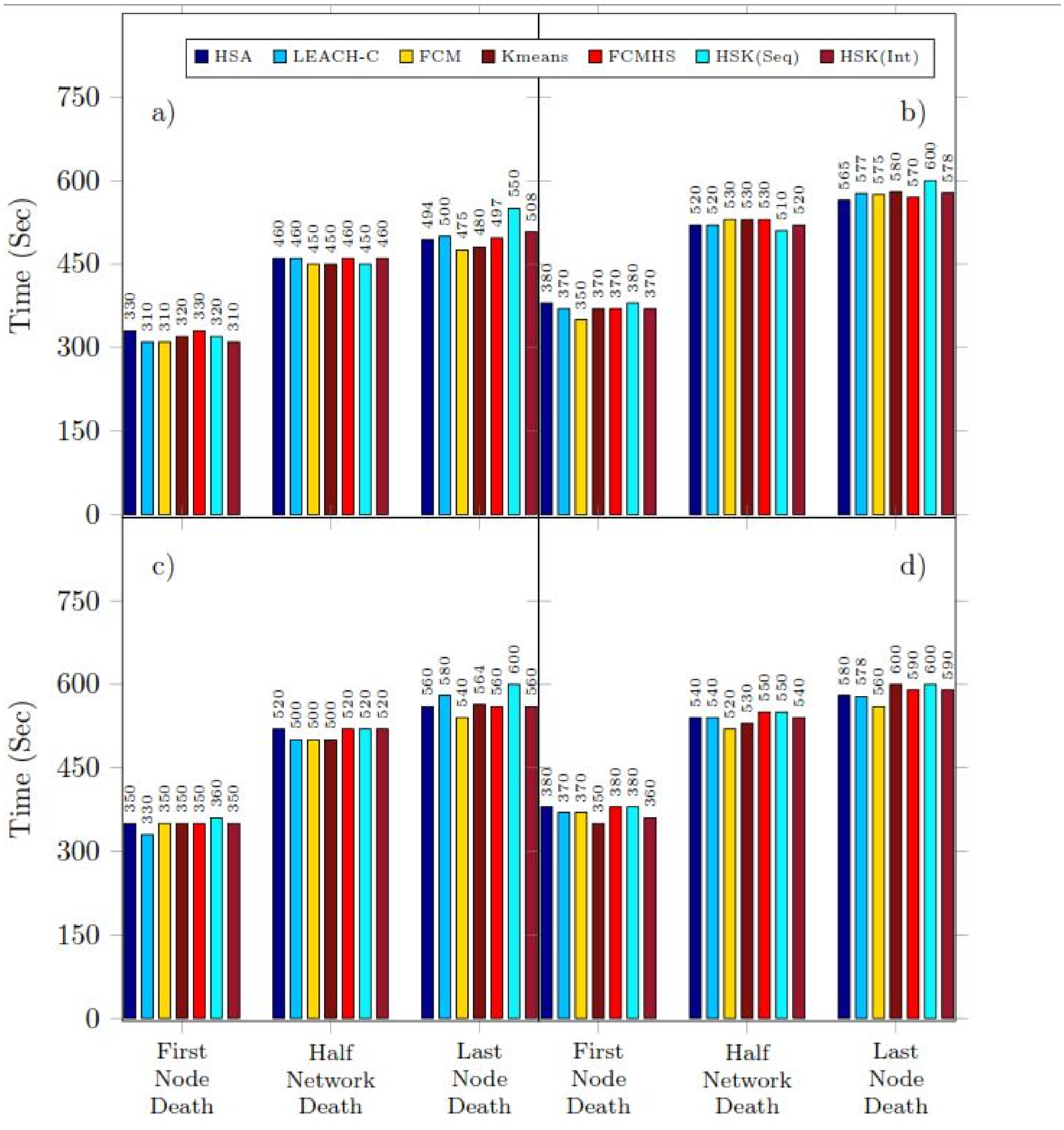


Figure 5.9: Alive nodes for all topology

5.1.6 Variable Improvisation for Harmony search

We have conducted a experiment for how the objective function of Harmony search is converging and getting best harmony value from available in Harmony memory.

So we have simulated the same code for 10,20,30,50,70,100 number of cycles for improvisation to check how the values of best harmony taken is varying.

In this section the graph plotted at 5.10 shows about how the best harmony is selected.Here we can see that when we increase total number of cycles of improvisation, the random exploration is increased, so more optimal values we get, and because of more optimisation the value of objective function for best chose harmony will be less(because it is minimisation). We can see that for higher number of cycles, both deviation and actual value of objective function is less compared to less number of cycle output.

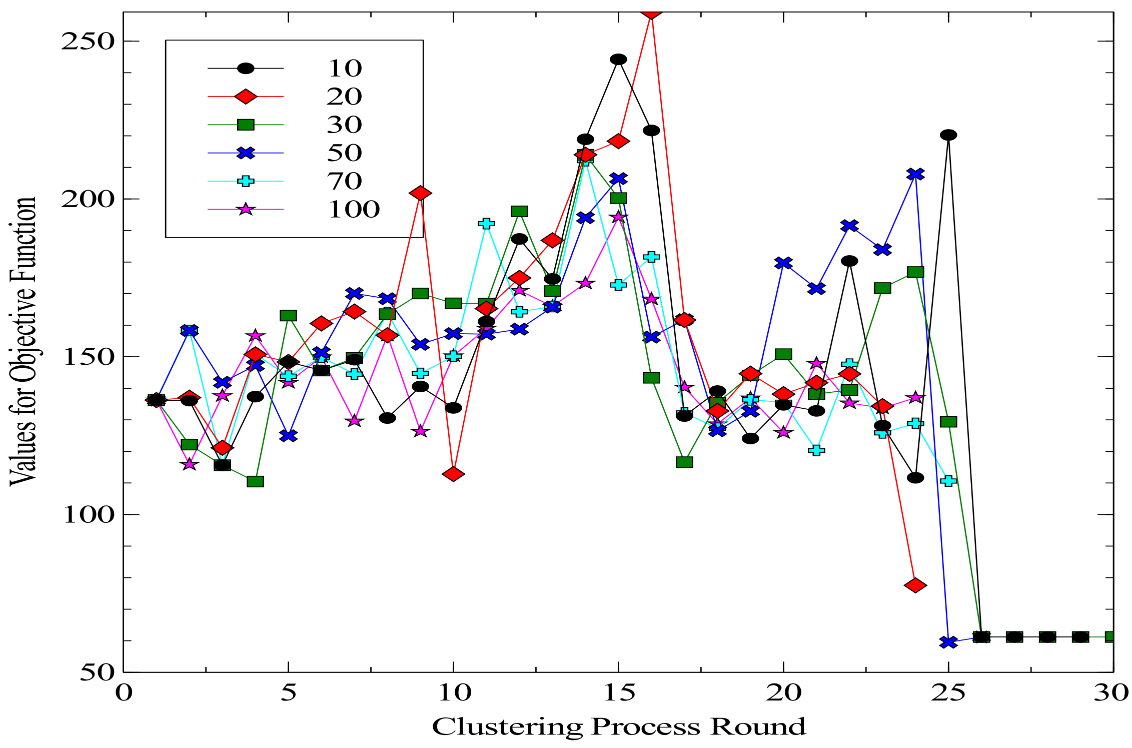


Figure 5.10: Variable Improvisation

5.1.7 Tables for Alive-nodes FND-HND-LND

In this section tables are shown for death of first node, death of half number of nodes and death of last node for different topologies like 50nodes, 100nodes, 150nodes and 200nodes.

Algorithms	**50nodes**		
	FND(Sec)	**HND(Sec)**	**LND(Sec)**
HSA	330	460	494
Leach-C	310	460	500
FCM	310	450	475
Kmeans	310	450	480
FCMHS	330	460	497
HS-Kmeans(seq)	320	450	540
HS-Kmeans(Int)	310	460	508

Table 5.1: Center-50nodes

Algorithms	**100nodes**		
	FND(Sec)	**HND(Sec)**	**LND(Sec)**
HSA	380	520	565
Leach-C	370	520	577
FCM	350	530	575
Kmeans	370	530	580
FCMHS	370	530	570
HS-Kmeans(seq)	380	510	600
HS-Kmeans(Int)	370	520	578

Table 5.2: Center-100nodes

Algorithms	**150nodes**		
	FND(Sec)	**HND(Sec)**	**LND(Sec)**
HSA	350	520	560
Leach-C	330	500	580
FCM	350	500	540
Kmeans	350	500	564
FCMHS	350	520	560
HS-Kmeans(seq)	360	520	600
HS-Kmeans(Int)	350	520	560

Table 5.3: Center-150nodes

Algorithms	**200nodes**		
	FND(Sec)	**HND(Sec)**	**LND(Sec)**
HSA	380	540	580
Leach-C	370	540	578
FCM	370	520	560
Kmeans	350	530	600
FCMHS	380	550	590
HS-Kmeans(seq)	380	550	600
HS-Kmeans(Int)	360	540	590

Table 5.4: Center-200nodes

5.1.8 Time taken for Clustring process

In the table 5.6 and 5.6time taken for clustering for different algorithms are displayed,where in first table the time displayed is CPU time and second is actual time according to our clock.

Algorithms	**CPU Time(millisec)**			
	50nodes	**100nodes**	**150nodes**	**200nodes**
HSA	46	319	1136	2333
Leach-C	9	30	73	161
FCM	46	275	583	1552
Kmeans	20	50	74	120
FCMHS	16	148	375	1095
HS-Kmeans(seq)	3	10	30	68
HS-Kmeans(Int)	5	22	61	120

Table 5.5: CPU Time Taken for clustering

Algorithms	**Actual Time(millisec)**			
	50nodes	**100nodes**	**150nodes**	**200nodes**
HSA	45	319	1140	2335
Leach-C	24	81	151	317
FCM	97	349	1042	1163
Kmeans	17	28	62	105
FCMHS	29	398	602	1769
HS-Kmeans(seq)	10	25	67	128
HS-Kmeans(Int)	13	46	131	252

Table 5.6: Actual Time Taken for clustering

5.2 Conclusion

In this thesis, three different hybrid methods namely 1) Harmony-search and K-means(sequential) 2) Harmony-search and K-means(Interleaved) and 3) Fuzzy C-means and Harmony search were proposed and in my research work, those approaches were simulated in NS2 simulator for different topologies. By the results of those simulations, I conclude that hybridization of meta-heuristic methods, done properly and efficiently, improves energy efficiency and network sustainability for longer time as compared to the individual methods results.

5.3 Future work

Many new algorithms like firefly algorithm, artificial bee colony, Dialectic Search and other evolutionary approaches are emerging right now, and their positive aspect and strengths can be evaluated to check how that can be hybridized with other conventional methods or meta-heuristic methods, to get the more optimal solutions.

Other than that, many well known algorithms are also not yet evaluated how they performs under certain hybrid conditions, that can be a future scope for having hybrid approaches which can outperform the single individual methods.

Chapter 6

List of Papers Published

1. *"Optimization of clustering process in WSN with meta-hueristic techniques - A survey"*, 3rd IEEE International Conference on Recent Advances in Information Technology, (RAIT 2016)Organized by ISM Dhanbad,March 2016.

2. *"Optimization of Clustering process for WSN with Hybrid Harmony search and K-means algorithm"*, 5th International Conference On Recent Trends In Information Technology, (ICRTIT 2016) Organized By MIT Anna University ,Sponsered by IEEE & ISRO.

Bibliography

[1] B. Solaiman and A. Sheta, "Energy optimization in wireless sensor networks using a hybrid k-means pso clustering algorithm," *Turkish Journal of Electric Engineering and Computer Science, Accepted for publications*, 2015.

[2] S. Vishwakarma, "An analysis of leach protocol in wireless sensor network: A survey,"

[3] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *System sciences, 2000. Proceedings of the 33rd annual Hawaii international conference on*, pp. 10–pp, IEEE, 2000.

[4] G. Y. Park, H. Kim, H. W. Jeong, and H. Y. Youn, "A novel cluster head selection method based on k-means algorithm for energy efficient wireless sensor network," in *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*, pp. 910–915, IEEE, 2013.

[5] H. Heidary, N. Z. Karimi, M. Ahmadi, A. Rahimi, and A. Zucchelli, "Clustering of acoustic emission signals collected during drilling process of composite materials using unsupervised classifiers," *Journal of Composite Materials*, vol. 49, no. 5, pp. 559–571, 2015.

[6] G. C. Pereira, M. M. de Oliveira, and N. F. Ebecken, "Genetic optimization of artificial neural networks to forecast virioplankton abundance from cytometric data," *Journal of Intelligent Learning Systems and Applications*, vol. 5, no. 1, p. 57, 2013.

[7] W. Sun, J. Wang, and H. Chang, "Forecasting annual power generation using a harmony search algorithm-based joint parameters optimization combination model," *Energies*, vol. 5, no. 10, pp. 3948–3971, 2012.

[8] B. Bollobás, *Random graphs.* Springer, 1998.

[9] M. Karimi, H. R. Naji, and S. Golestani, "Optimizing cluster-head selection in wireless sensor networks using genetic algorithm and harmony search algorithm," in *Electrical Engineering (ICEE), 2012 20th Iranian Conference on*, pp. 706–710, IEEE, 2012.

[10] T. Murata and H. Ishibuchi, "Performance evaluation of genetic algorithms for flowshop scheduling problems," in *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on*, pp. 812–817, IEEE, 1994.

[11] P. Sasikumar and S. Khara, "K-means clustering in wireless sensor networks," in *Computational intelligence and communication networks (CICN), 2012 fourth international conference on*, pp. 140–144, IEEE, 2012.

[12] D. Hoang, R. Kumar, and S. Panda, "Fuzzy c-means clustering protocol for wireless sensor networks," in *Industrial Electronics (ISIE), 2010 IEEE International Symposium on*, pp. 3477–3482, IEEE, 2010.

[13] M. Dorigo and M. Birattari, "Ant colony optimization," in *Encyclopedia of machine learning*, pp. 36–39, Springer, 2010.

[14] J. Du and L. Wang, "Uneven clustering routing algorithm for wireless sensor networks based on ant colony optimization," in *Computer Research and Development (ICCRD), 2011 3rd International Conference on*, vol. 3, pp. 67–71, IEEE, 2011.

[15] M. Ziyadi, K. Yasami, and B. Abolhassani, "Adaptive clustering for energy efficient wireless sensor networks based on ant colony optimization," in *Communication Networks and Services Research Conference, 2009. CNSR'09. Seventh Annual*, pp. 330–334, IEEE, 2009.

[16] M. Mitchell, *An introduction to genetic algorithms.* MIT press, 1998.

[17] S. Hussain, A. W. Matin, and O. Islam, "Genetic algorithm for energy efficient clusters in wireless sensor networks," in *null*, pp. 147–154, IEEE, 2007.

[18] Z. W. Geem, J. H. Kim, and G. Loganathan, "A new heuristic optimization algorithm: harmony search," *Simulation*, vol. 76, no. 2, pp. 60–68, 2001.

[19] D. Hoang, P. Yadav, R. Kumar, and S. Panda, "A robust harmony search algorithm based clustering protocol for wireless sensor networks," in *Communications Workshops (ICC), 2010 IEEE International Conference on*, pp. 1–5, IEEE, 2010.

[20] Y. Huang, S. Zheng, P. Zhang, and X. Qin, "Adaptive genetic simulated annealing algorithm for wsns," in *Computer Engineering and Technology (ICCET), 2010 2nd International Conference on*, vol. 4, pp. V4–328, IEEE, 2010.

[21] R. V. Kulkarni, "Senior member, ieee, ganesh kumar, venayagamoorthy, senior member, ieee,particle swarm optimization in wireless sensor networks: A brief survey," *IEEE Transactions on Systems, Man and Cybernetics-Part C: Applications and Reviews*, vol. 41, no. 2, 2011.

[22] N. Latiff, C. C. Tsimenidis, and B. S. Sharif, "Energy-aware clustering for wireless sensor networks using particle swarm optimization," in *Personal, Indoor and Mobile Radio Communications, 2007. PIMRC 2007. IEEE 18th International Symposium on*, pp. 1–5, IEEE, 2007.

[23] A. A. Baraa and E. A. Khalil, "A new evolutionary based routing protocol for clustered heterogeneous wireless sensor networks," *Applied Soft Computing*, vol. 12, no. 7, pp. 1950–1957, 2012.

[24] E. A. Khalil and A. A. Baraa, "Energy-aware evolutionary routing protocol for dynamic clustering of wireless sensor networks," *Swarm and Evolutionary Computation*, vol. 1, no. 4, pp. 195–203, 2011.

[25] A. Sheta and B. Solaiman, "Evolving a hybrid k-means clustering algorithm for wireless sensor network using pso and gas," *International Journal of Computer Science Issues (IJCSI)*, vol. 12, no. 1, p. 23, 2015.

[26] L. Tan, Y. Gong, and G. Chen, "A balanced parallel clustering protocol for wireless sensor networks using k-means techniques," in *Sensor Technologies and Applications, 2008. SENSORCOMM'08. Second International Conference on*, pp. 300–305, IEEE, 2008.

[27] T. Camilo, J. S. Silva, A. Rodrigues, and F. Boavida, “Gensen: A topology generator for real wireless sensor networks deployment,” in *Software Technologies for Embedded and Ubiquitous Systems*, pp. 436–445, Springer, 2007.