CHAPTER 4

AN EXPONENTIAL CUCKOO SEARCH FOR ENERGY EFFICIENT CLUSTERING IN WIRELESS SENSOR NETWORK

4.1 INTRODUCTION

Sensor node are densely deployed in WSN that means physical environment would produce very similar data in close by sensor node and transmitting such type of data is more or less redundant. So all these facts encourage using some kind of grouping of sensor nodes such that group of sensor node can combine or compress data together and transmit only compact data. This can reduce localized traffic in individual group and also reduce global data .this grouping process of sensor nodes in a densely deployed large scale sensor node is known as clustering. The way of combing data and compress data belonging to a single cluster called data aggregation (Maraiya et al 2011).

Issues of clustering in WSN: (a) how many sensor nodes should be taken in a single cluster. Selection procedure of cluster head in an individual cluster.(b) Heterogeneity in a network, it means user can put some power full nodes, in term of energy in the network which can behave like cluster head and simple node in a cluster work as a cluster member only. Many protocol and algorithm have been proposed which deal with each individual issue.

4.1.1 Issues to be considered in Clustering

The clustering phenomenon, plays an important role in not just organization of the network, but can dramatically affect network performance.

There are several key limitations in WSNs, that clustering schemes must consider (Janani et al 2013).

Limited Energy: Unlike energetic designs, wireless sensor nodes are off-grid, meaning that they have limited energy storage and the efficient use of this energy will be vital in determining the range of suitable applications for these networks. The limited energy in sensor nodes must be considered as proper clustering can reduce the overall energy usage in a network.

Network Lifetime: The energy limitation on nodes results in a limited network lifetime for nodes in a network. Proper clustering should attempt to reduce the energy usage, and hereby increase network lifetime.

Limited Abilities: The small physical size and small amount of stored energy in a sensor node limit many abilities of nodes in terms of processing and communication abilities. A good clustering algorithm should make use of shared resources within an organizational structure, while taking into account the limitation on individual node abilities.

Application Dependency: Often a given application will heavily rely on cluster organization. When designing a clustering algorithm, application robustness must be considered as a good clustering algorithm should be able to adapt to a variety of application requirements.

4.2 FACTORS CONSIDERED DURING CLUSTER HEAD SELECTION

A Density and Distance based Cluster Head Selection (DDCHS) Algorithm in Sensor Networks: The each steps of DDCHS algorithm are described as follows (Lee et al 2010)

- Local Grouping divides cluster area into two perpendicular diameters to get four quadrants.
- Compare the node density that is the number of cluster members in each quadrant and select candidate quadrants.
- Compare the node distance that is from the nearest cluster head in candidate quadrants and select cluster head.

An Energy Efficient Algorithm for Cluster-Head Selection in WSNs: There are various network models for WSNs. For the development of our method, some reasonable assumptions about the sensor nodes are made. These assumptions are similar to those incorporated in and are as follow (Hajikhani & Abolhassani 2010):

- All the sensor nodes are homogeneous, stationary and energy constrained.
- The base station is a high-energy node, located far away from the sensor nodes.
- All the nodes can transmit with enough power to reach to the base station if needed.
- Nodes always have data to send to the end user and nodes located close to each other, have correlated data.

Consumed Energy as a Factor for Cluster Head Selection in Wireless Sensor Networks: The inclusion of consumed energy, a new approach, as extended factor to reduce threshold increases Life time better than residual energy. The proposed formula is (Raj 2012):

$$T(n) = \begin{cases} \dfrac{P^* \tan(E_{out} / E_{in})}{1 - p^* (r^* \bmod (1/p))}; & if \ \ n \in G \\ 0 \qquad\qquad\qquad ; & Otherwise \end{cases}$$

- It enables to reduce the threshold T (n) without network stuck.
- Nodes with less consumed energy will more likely become cluster heads than nodes with more consumed energy in a given round.
- With longer distance between the base station and the nodes, network lifetime has no significant change.
- It can be achieved a remarkable increase in network life time.

4.3 WHY CLUSTER HEAD SELECTION IS NP-HARDNESS

Optimization is a term that covers almost all sectors of human life and work; from scheduling of airline routes to business and finance, and from wireless routing to engineering design. In fact, almost all research activities in computer science and engineering involve a certain amount of modelling, data analysis, computer simulations, and optimization. In a word, it is an applied science that tries to obtain the related parameter values which facilitate an objective function to produce some minimum or maximum value. In the real world, resources are limited, time and money are always less than required, so optimization is far more important in practice (Adnan et al 2013).

A typical optimization process consists of three components: model, optimizer and simulator. The representation of the physical problem is done by using mathematical equations which can be converted into a numerical model. The formulation of a simple optimization problem can be done in many ways.

For instance, the most popular way to do the formulation is to write a nonlinear optimization problem as:

$$\min imize\ f_i(x),\ (i = 1, 2, ..., M)$$

Subject to the constraints:

$$h_j(x),\ (j = 1, 2, ..., J)$$

$$g_k(x),\ < 0\ (\mathrm{k} = 1, 2, ..., \mathrm{K})$$

Where f_i, h_i *and* g_k are nonlinear functions. Here the design vector $x = (x_1, x_2, ...)$ can be continuous, discrete or mixed in n-dimension. The function f_i is called objective function (cost function). Here when M is 1, it is a single objective function. But when M > 1, the optimization is multi objective. It is possible to combine different objectives into a single objective and in some cases it is a useful approach. It can be noted that the problem it formulated here is a minimization problem. The maximization problem can be written by simply substituting $f_i(\mathrm{x})$ by $-f_i(\mathrm{x})$.

Cluster-head dissipates more energy and if it remains cluster-head permanently it will die quickly as happened in case of static clustering. Low-power optimization techniques developed for conventional ad hoc networks are not sufficient as they do not properly address particular features of embedded and sensor networks. It is not enough to reduce overall energy consumption, it is also important to maximize the lifetime of the entire network, that is, maintain full network connectivity for as long as possible (Aslam et al 2012).

Network optimization is a critical component and the optimization techniques are used to achieve design goals in networking. Energy efficiency, cost and application requirement are the challenges that are to be taken care while designing a WSN. This requires optimization of both hardware and software to make WSN efficient. Software addresses issue of Network Lifetime. There are several optimization algorithms to suit the different problems. Choosing a proper algorithm is very important in any optimization technique. Efficient use of battery energy is hence crucial to enhance the network lifetime (Bilouhan & Gupta 2011).

The sensor network protocols have to focus primarily on power conservation issues. Other issues are, achieving high quality QoS, low bandwidth, limited processing and storage in every node. These are the issues which are directly related to the problem of optimization (Honguntikar & Biradar 2014).

A meta-heuristic algorithm is an iterative generation process which combines different concepts for exploring the search space and finds near-optimal solutions. These algorithms are approximate and usually nondeterministic. Meta-heuristic algorithms are best suitable for NP-hard optimization problems. They give better quality solutions than heuristic methods. Query optimization is an NP-hard problem. As an alternate to the traditional optimization methods, it can optimize the query by using meta-heuristic algorithms which are the best choice for solving NP-hard problems (Gomathi & Sharmila 2014).

4.4 METHODOLOGY

Cuckoo Search (CS) an optimization algorithm developed by Xin-She Yang and Susah Deb in 2009 is a new algorithm inspired by the brood

parasitism of some cuckoo species which lay their eggs in nests of other species host birds. It is more efficient than Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) and adapts to a wider class of optimization problems (Yang and Deb 2014). In this work, load Balanced CS based on residual energy and path length search for multiple paths between cluster head and sink node.

CS is heuristic search algorithm inspired by the cuckoo's reproduction strategy. Nature inspired computation techniques are computing techniques derived from natural system study. Candidate solutions for optimization problem play the role of population individuals and fitness function determines the solutions quality. Commonly used nature inspired algorithms are GA, Firefly, Artificial bee colony, Ant Colony Optimization (ACO), CS, Bat and PSO.

Generally, cuckoo eggs hatch earlier than host eggs. Once a first cuckoo chick comes out, the immediate instinctive action it takes is to evict host eggs by blindly propelling them out of the nest, thereby increasing the cuckoo chick's share of food from the host bird. Each egg represents a solution with a cuckoo egg representing a new solution. The aim is to employ new and potentially better solutions (cuckoos) replacing not-so-good solutions in nests (Sakthi & Nedunchezhian 2014).

In a simple form, each nest has an egg. The algorithm is capable of being extended to more complicated cases where each nest has many eggs representing a solutions set. CS is based on three idealized rules (Valian et al 2011):

- Each cuckoo lays one egg at a time, and dumps it in a randomly chosen nest;

- The best nests with high quality of eggs (solutions) will carry over to the next generations;
- The number of available host nests is fixed, and a host can discover an alien egg with probability $p_a \in [0, 1]$. In this case, the host bird can either throw the egg away or abandon the nest to build a completely new nest in a new location.

CS algorithm is based on the brood parasitic behaviour of cuckoo species joined with the Levy flight behaviour of birds and fruit flies. Some species of Cuckoos lay eggs in communal nests. When a host bird discovers the eggs are not its own, it either throws the alien eggs away or abandons its nest and builds a new one elsewhere. CS, is described using three idealized rules: a) a cuckoo lays one egg at a time, and dump it in a randomly chosen nest; b) The best nests with high quality eggs will carry over to the next generation; c) number of available host nests is fixed, and the cuckoo egg is discovered by the host births a probability pa ϵ [0, 1]. Worst nests are discovered and dumped from further calculations.

Generalized Pseudo code of the CS (Valian et al 2011):

begin
Objective function $f(X)$, $X = (x1, ..., xd)$
Generate initial population of n host nests $X(i = 1, 2, ..., n)$
while $(t < MaxGeneration)$ *or* $(stop\ criterion)$
Get a cuckoo randomly by Levy flights
evaluate its quality / fitness F_i
Choose a nest among n (say, j) *randomly*
if $(F_i > F_j)$,
replace j by the new solution;
end
A fraction (p_a) *of worse nests is abandoned and new ones are built*;
Keep the best solutions $(or\ nests\ with\ quality\ solutions)$;
Rank the solutions and find the current best
end while
Postprocess results and visualization
end

Quality or solution fitness is proportional to value of objective function for optimization problem.

X is related to a new solution, for an i^{th} cuckoo and then levy is performed as in Equations:

$$X_i^{(t+1)} = X_i^{(t)} + \alpha * Levy(\tilde{n})$$

$$X_i^{(t+1)} = X_i^{(t)} + \alpha * E_t$$

In WSN cluster formation, nodes are randomly chosen as cluster head Initial population of nest is represented by nodes where c represents a node selected as a cluster head for that initial random solution. For a dimension d in search space c_{id}^k indicates a node is placed for nest *i* in period *d* at iteration *k.* In other words, c_{id}^k is a binary value so that c_{id}^k =1if a node is

selected as cluster head else $c_{id}^{k} = 0$ otherwise. During setup phase a node becomes a cluster head with a probability of 0.5. Specifically, if $p(0,1) > 0.5$ then $c_{id}^{0} = 1$ else $c_{id}^{0} = 0$.

When clusters are formed and cluster head elected during setup phase, route to Base Station from a cluster head is determined. With cluster formation further multiple cluster formation based on available cluster heads and distance between cluster head and sink is performed.

Load Balanced CS based on residual energy and path length is applied to search multiple paths between Cluster Head and sink node. The Iterative deepening depth-first search approach this as follows: First, a depth-first search for a node is performed. Then, discarding nodes generated in first search, it starts over and does a depth-first search to level two. Next, start again and do a depth-first search to depth three and so on, continuing till a goal state is reached. Iterative deepening depth-first search approach expands all nodes at a depth before expanding nodes at greater depth finding multiple paths.

4.4.1 Procedure for Cuckoo Search Algorithm

1. A set of N nest are randomly generated in which the nest positions are coordinated by the sensors. Initialization is performed for number of sensor nodes, cuckoo nests, eggs in nests, location and energy of nodes and base station.

2. Using CS, clusters are formed where each egg in the nest corresponds to a sensor node. M nest group is chosen with N eggs present in it. Randomly choosing of best or quality eggs

are done by doing step size and levy angle updating at each iteration thereby nest gets updated.

3. The best egg is considered as cluster head since it includes high energy node in context to energy, distance between the node and distance to the base station. Where areas worst nest are abandoned in normal CS.

4. At each run, based on residual energy of the nodes, cluster heads changes periodically which is used to eliminate the communication overhead and redundancy.

The Figure 4.1 shows the flowchart of CS (Gomathi & Sharmila 2014):

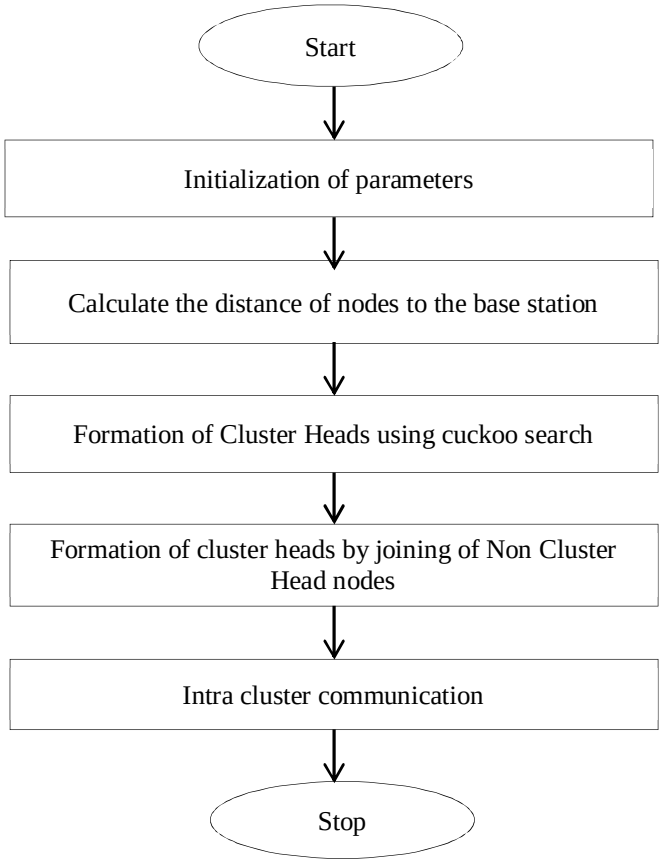

Figure 4.1 Flowchart of cuckoo search

4.5 RESULTS AND DISCUSSION

Simulations were carried out by varying the number of nodes from 50 to 300. In this study, Table 4.2 to 4.5 and Figure 4.2 to 4.5 shows result of number of clusters formed, average end to end delay, average packet loss rate and lifetime computation respectively. Table 4.1 shows the simulation setup.

Table 4.1 Simulation Setup

Number of nodes	50, 100, 150, 200, 250, 300
Number of Cuckoos	15
Size	30x30,40x40
Probability	Pa ∈ [0, 1]

Table 4.2 Number of clusters formed

Number of nodes	**LEACH**	**Cuckoo search**
50	9	10
100	13	16
150	23	25
200	29	29
250	29	31
300	33	37

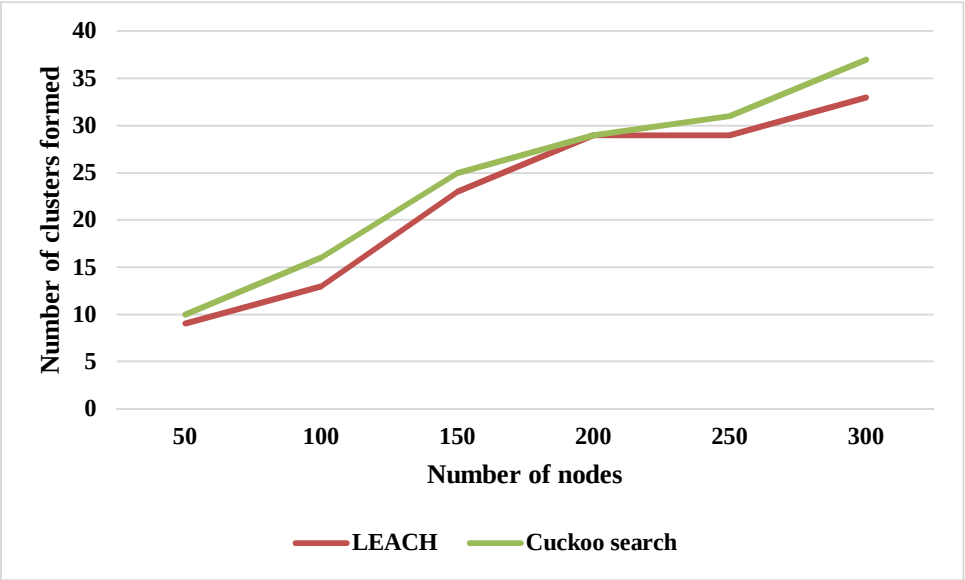

Figure 4.2 Number of clusters formed

Figure 4.2 explains that the cuckoo search performs better in cluster formation when compared to LEACH. The performance of cuckoo search is better in the range of 0 to 20.68% than LEACH.

Table 4.3 Average End to End Delay (sec)

Number of nodes	LEACH	Cuckoo search
50	0.00138	0.00132
100	0.00141	0.00137
150	0.01407	0.01342
200	0.02296	0.02245
250	0.04851	0.04814
300	0.05621	0.04989

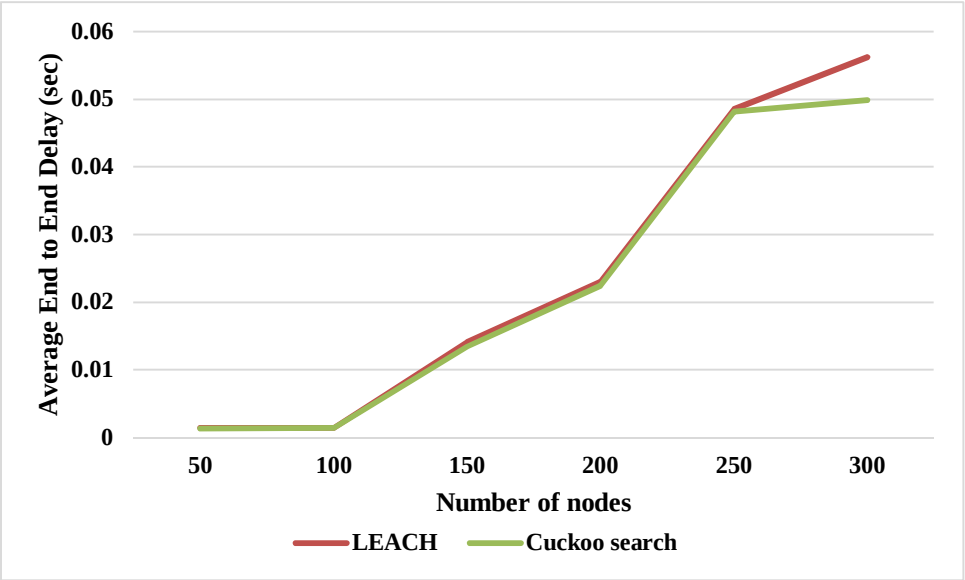

Figure 4.3 Average End to End Delay (sec)

Figure 4.3 explains that the cuckoo search performs better in reducing the delay when compared to LEACH. The performance of cuckoo search is better in the range of 0.76% to 11.9% than LEACH.

Table 4.4 Average Packet loss rate (%)

Number of nodes	LEACH	Cuckoo search
50	9.37	7.88
100	15.05	12.44
150	15.44	12.26
200	20.57	18.37
250	27.68	23.89
300	39.14	28.54

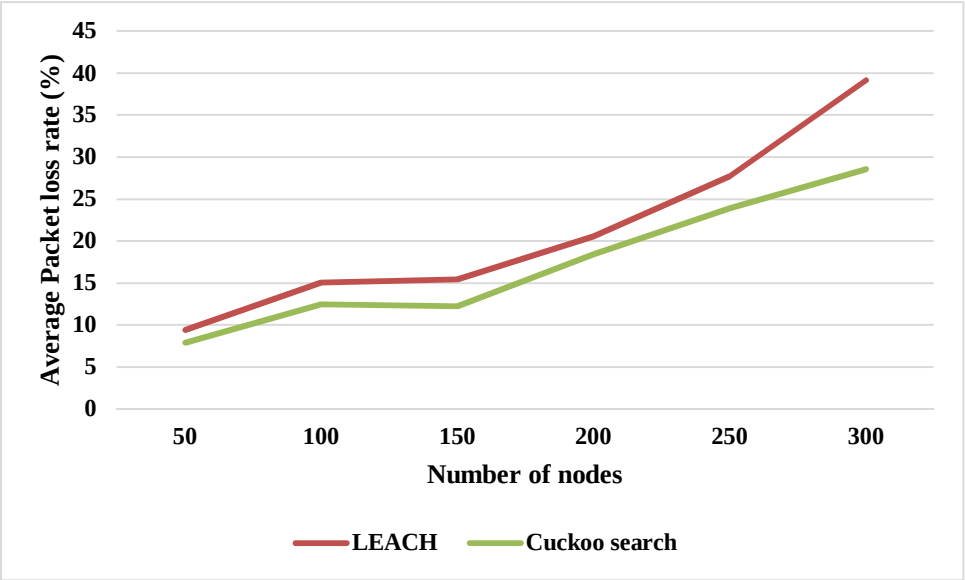

Figure 4.4 Average Packet loss rate (%)

Figure 4.4 explains that the cuckoo search performs better in reducing the packet loss when compared to LEACH. The performance of cuckoo search is better in the range of 11.29% to 31.32% than LEACH.

Table 4.5 Lifetime Computation

Number of rounds	LEACH	Cuckoo Search
0	100	100
100	100	100
200	87	96
300	74	91
400	41	77
500	24	56
600	2	27
700	0	9
800	0	0

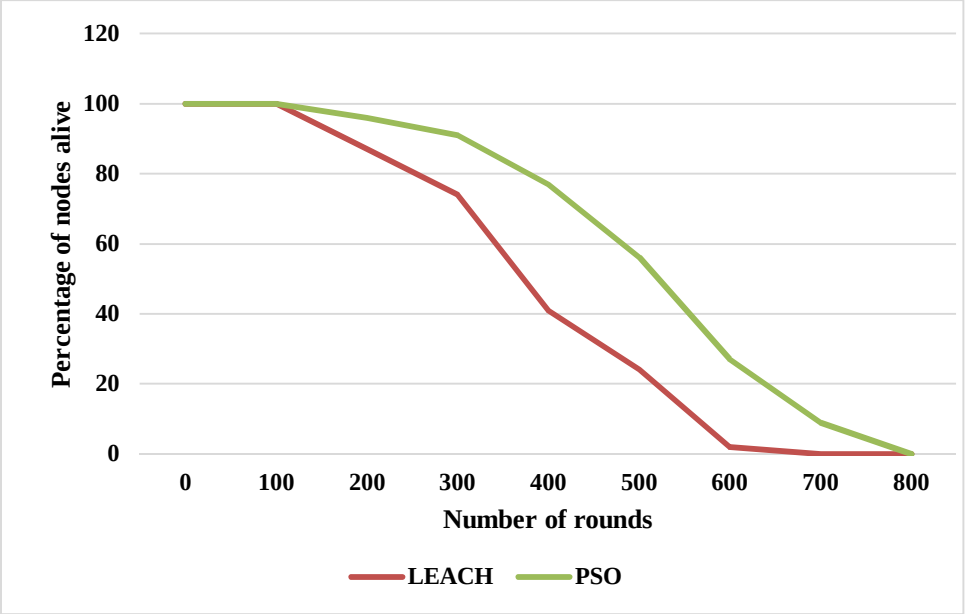

Figure 4.5 Lifetime computation

Figure 4.5 explains that the cuckoo search performs better in lifetime computation when compared to LEACH. The performance of cuckoo search is better in the range of 0 to 172.41% than LEACH.

4.6 CONCLUSION

A hybrid CS algorithm is presented. A residual energy and path length based Load Balanced CS is applied to search for multiple paths between Cluster Head and sink node. A load balancing function is suggested to distribute traffic over discovered multiple paths. Cluster Head selection is through using CS. Suboptimal solutions found during Cluster Head are key nodes for multiple routes formation using Iterative deepening depth-first search approach. Results show that cuckoo search performs better in cluster formation compared to LEACH in a range of 0 to 20.68%. CS performs better than LEACH by reducing delay and packet loss rate.