# CHAPTER 5

# A MODIFIED CUCKOO SEARCH FOR ENERGY EFFICIENT CLUSTERING IN WIRELESS SENSOR NETWORK

## 5.1 INTRODUCTION

The process of optimization is searching a vector in a function that produces an optimal solution. All of feasible values are available solutions, and the extreme value is optimal solution. In general, optimization algorithms are applied to solve these optimization problems. A simple classification way for optimization algorithms is considering the nature of the algorithms, and optimization algorithms can be divided into two main categories: deterministic algorithms and stochastic algorithms. Deterministic algorithms using gradients such as hill climbing have a rigorous move and will generate the same set of solutions if the iterations commence with the same initial starting point (Wang & Guo 2013).

On the other hand, stochastic algorithms without using gradients often generate different solutions even with the same initial value. However, generally speaking, the final values, though slightly different, will converge to the same optimal solutions within a given accuracy. Generally, stochastic algorithms have two types: heuristic and meta-heuristic. Recently, nature-inspired meta-heuristic algorithms perform powerfully and efficiently in solving modern nonlinear numerical global optimization problems. To some extent, all meta-heuristic algorithms strive for making balance between randomization (global search) and Local Search.

A new class of optimization heuristics, hybrid optimizations. Hybrid optimizations assume that one has implemented two or more algorithms for the same optimization. A hybrid optimization uses a heuristic to choose the best of these algorithms to apply in a given situation. Here to construct a hybrid register allocator that chooses between two different register allocation algorithms, graph coloring and linear scan (Cavazos et al 2006). The goal is to create an allocator that achieves a good balance between two factors: trying to find a good packing of the variables to registers (and thereby achieving good running time performance) and trying to reduce the overhead of the allocator.

A hybrid optimization will reduce compilation effort, using an efficient algorithm most of the time, but will use a more effective, but expensive, optimization algorithm seldomly, when it deems the additional benefit is worth the effort.

A hybrid optimization approach for the design of linkages method is applied to the dimensional synthesis of mechanism and combines the merits of both stochastic and deterministic optimization. The stochastic optimization approach is based on a real-valued Evolutionary Algorithm (EA) and is used for extensive exploration of the design variable space when searching for the best linkage (Sedano et al 2012).

The deterministic approach uses a local optimization technique to improve the efficiency by reducing the high CPU time that EA techniques require in this kind of applications. To that end, the deterministic approach is implemented in the EA in two stages. The first stage is the fitness evaluation where the deterministic approach is used to obtain an effective new error estimator. In the second stage the deterministic approach refines the solution provided by the evolutionary part of the algorithm. The new error estimator

enables the evaluation of the different individuals in each generation, avoiding the removal of well-adapted linkages that other methods would not detect.

## 5.2 PROBLEM FORMULATION

The energy consumption model is called first order radio model. Each node consumes the following $E_{Tx}$amount of energy to transmit an l-bits message over distance d and the $E_{Rx}$amount of energy to receive the message

$$E_{Tx}(l,d)=\begin{cases} lE_{elec}+l\varepsilon_{fs}d^2, d<d_0 \\ lE_{elec}+l\varepsilon_{mp}d^4, d\geq d_0 \end{cases};E_{Rx}(l)=lE_{elec}$$

According to first order radio model, each cluster consumes the $E_{cluster}$ amount of energy to communicate with Sensor Node. To minimize the total energy consumption *total E*, to get the optimal cluster number $k_{opt}$and the formula is shown as the follows (Wang et al., 2002)

$$E_{cluster}=E_{CH}+(\frac{N}{k}-1)E_{member}\approx lE_{elec}(\frac{N}{k}-1)+E_{Tx}+(lE_{elec}+l\varepsilon_{fs}\frac{R^2}{2k})\frac{N}{k}$$

$$E_{total}=kE_{cluster}=\mathrm{k}E_{CH}+NE_{member}=2NlE_{elec}-lkE_{elec}+KE_{Tx}+Nl\varepsilon_{fs}\frac{R^2}{2k}$$

$$k_{opt}=\frac{R}{d^2}\sqrt{\frac{N\varepsilon_{fs}}{2\varepsilon_{mp}}}$$

## 5.3 METHODOLOGY

Energy efficiency, cost and application requirement are the challenges that are to be taken care while designing a WSN. So optimization is performed using CS. For that purpose it propose modified cuckoo search algorithm.

### 5.3.1 Modified Cuckoo Search

In the real world, if a cuckoo's egg is very similar to a host's eggs, then this cuckoo's egg is less likely to be discovered, thus the fitness should be related to the difference in solutions. Therefore, it is a good idea to do a random walk in a biased way with some random step sizes. Both, original, and modified code use random step sizes. Compared to the original code, different function set are used for calculating this step size. In the original code, step size is calculated using below equation code expression (Tuba et al 2011):

$$r * nests[permute1[i]][j] - nests[permute2[i]][j]$$

where $r$ is random number in [0,1] range, *nests* is matrix which contains candidate solutions along with their parameters, *permute1* and *permute2* are different rows permutation functions applied on *nests* matrix. In order to calculate the step size, the modified CS uses the equation:

$$(rand() * a. \wedge pi)$$

Higher fitness solutions have slight advantage over solutions with lower fitness. This method keeps the selection pressure (the degree to which highly fit solutions are selected) towards better solutions and algorithm should achieve better results. That does not mean that high fitness solutions will flood population and the algorithm will stuck in local optimum.

### 5.3.2 Adaptive Cuckoo Search Algorithm (ACSA)

For any optimization approach, finding the optimum solutions competently and accurately relies utterly on the inherent search process. The effectiveness of the standard CSA is unquestionable, meaning that when

given enough computation time, it is guaranteed to converge to the optimum solutions eventually. However, the search process may be time consuming, due to the associated random walk behavior.

In order to improve the convergence rate while maintaining the eye-catching characteristics of the CSA, an accelerated searching process which, similarly to the inertia weight control strategy in the PSO, is proposed here (Li et al 2012).

The step size $\alpha$, which manages the local and global searching, is assigned as constant in the standard CSA, where $\alpha$ = 1 is applied. In this present work, a new ACSA is presented. Instead of using a constant value, the step size $\alpha$ is adjusted adaptively in the proposed ACSA, based on the assumption that the cuckoos lay their eggs at the area with a higher egg survival rate. In this regard, by adjusting the step size $\alpha$ adaptively, the cuckoos search around the current good solutions for laying an egg as this region probably will contain the optimal solutions, and, on the contrary, they explore more rigorously for a better environment if the current habitat is not suitable for breeding. The flow of the ACSA is given in Algorithm (Walton et al 2011).

*begin*

*Generate initial population of* $q$ *host nest* $x_i$, $i = 1, 2, \ldots, q$

*Define* min*imum step size value* $\alpha_L$

*Define* max*imum step size value* $\alpha_U$

*for all* $x_i$ *do*

*Evaluate the fitness function* $F_i = f(x_i)$

*end for*

*while* (*iter* < *MaxGeneration*) *or* (*stopping criterion*)

*Calculate the average fitness value* $F_{avg}$ *of all the host nests*

*Find the* min*imum fitness value* $F_{\min}$ *among all the host nests*

*for all host nests do*

*Current position* $x_i$

*Calculate the step size* $\alpha$ *for Levy flight u*sin*g*

$$\alpha = \begin{cases} \alpha_L + (\alpha_U - \alpha_L)\dfrac{F_j - F_{\min}}{F_{avg} - F_{\min}}, & F_j \le F_{avg} \\ \dfrac{\alpha_U}{\sqrt{t}}, & F_j > F_{avg} \end{cases}$$

*Generate a cuckoo egg* $x_j$ *from host nest* $x_i$ *by u*sin*g Levy flight*

*if* ($x_j$ *falls outside the bounds*) *then*

Re *place* $x_j$ *with* $x_i$

*end if*

*Calculate the fitness function* $F_j = f(x_j)$

*if* ($F_j > F_i$) *then*

Re *place* $x_i$ *with* $x_j$

Re *place* $F_i$ *with* $F_j$

*end if*

*end for*

*Abandon a fraction* $p_a$ *of the worst nests*

*Build new nests randomly to replace nests lost*

*Evaluate the fitness of new nests*

*end while*

*end*

The step size $\alpha$ determines how far a new cuckoo egg is located from the current host nest. Specifying the minimum and the maximum Levy

flight step size values properly is crucial such that the search process in the ACSA is neither too aggressive nor too ineffective. The $\alpha_L$ and $\alpha_M$ are chosen based on the domain of $x_i$. Additionally, as precautionary measure, if the ACSA generates a cuckoo egg that falls outside the domain of interest, its position will remain unchanged.

## 5.4 EXPERIMENTAL SETUP AND RESULTS

Experiments were conducted with 100 nodes with each node having a transmission capability of 100 meter.

**Table 5.1 Summary of Results for proposed Modified Cuckoo Search**

| **Number of clusters formed** | | | |
|---|---|---|---|
| **Number of nodes** | **LEACH** | **Cuckoo search** | **Modified Cuckoo Search** |
| 50 | 9 | 10 | 11 |
| 100 | 13 | 15 | 18 |
| 150 | 23 | 24 | 27 |
| 200 | 29 | 29 | 31 |
| 250 | 29 | 30 | 34 |
| 300 | 33 | 34 | 40 |
| **Average End to End Delay (sec)** | | | |
| 50 | 0.00138 | 0.00134 | 0.00124 |
| 100 | 0.00141 | 0.00142 | 0.00128 |
| 150 | 0.01407 | 0.01395 | 0.01229 |
| 200 | 0.02296 | 0.02274 | 0.0208 |
| 250 | 0.04851 | 0.04992 | 0.04627 |
| 300 | 0.05621 | 0.05172 | 0.04755 |
| **Average Packet loss rate (%)** | | | |
| 50 | 9.37 | 7.65 | 7.55 |
| 100 | 15.05 | 11.81 | 11.41 |
| 150 | 15.44 | 11.99 | 11.37 |
| 200 | 20.57 | 17.98 | 16.85 |
| 250 | 27.68 | 23.08 | 22.19 |
| 300 | 39.14 | 27.19 | 27.66 |

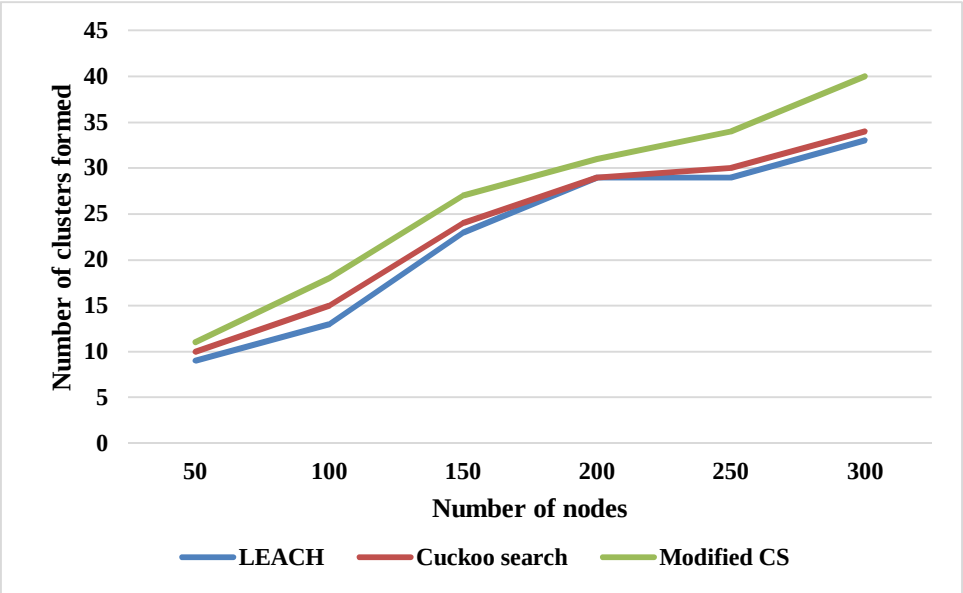


**Figure 5.1 Number of clusters formed**

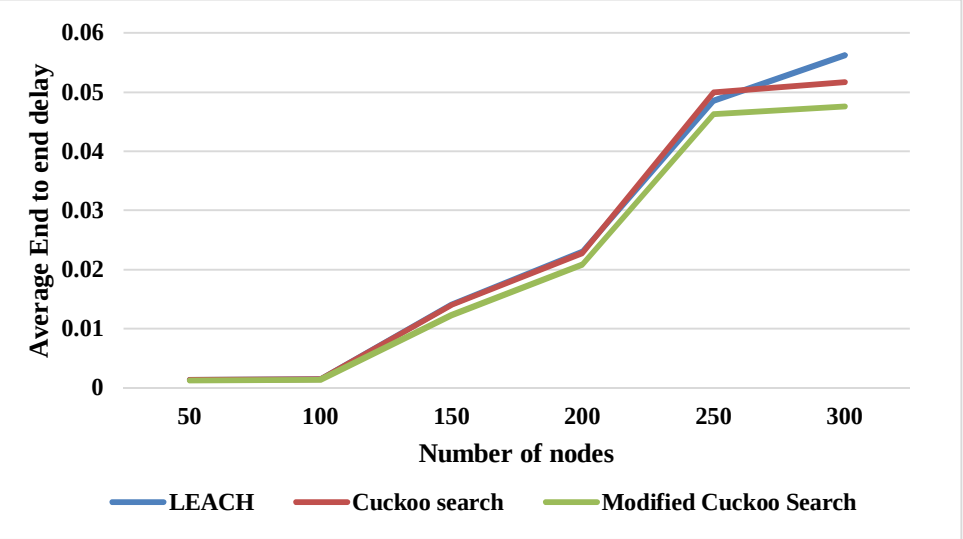


**Figure 5.2 Avg End to end delay**

The network is of size 100 sq m with the location of the Base station 50 meter away from the location (0, 0). The experiments conducted with LEACH, CS and modified CS methods. The lifetime was computed with different number of rounds. The results compared with one another. The proposed modified CS increased the number of clusters formed by 32.2581%

when compared with LEACH protocol with 100 nodes and by 16.2162% than CS with 300 nodes.

The modified CS reduced end to end delay averagely by 10.9409% when compared with LEACH and averagely by 8.4257% when compared with CS.

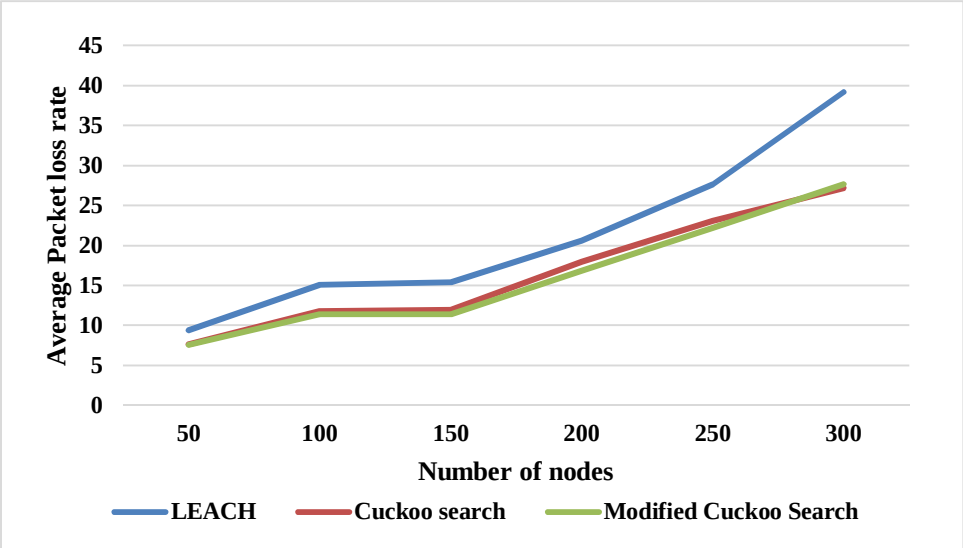


**Figure 5.3 Avg Packet loss rate**

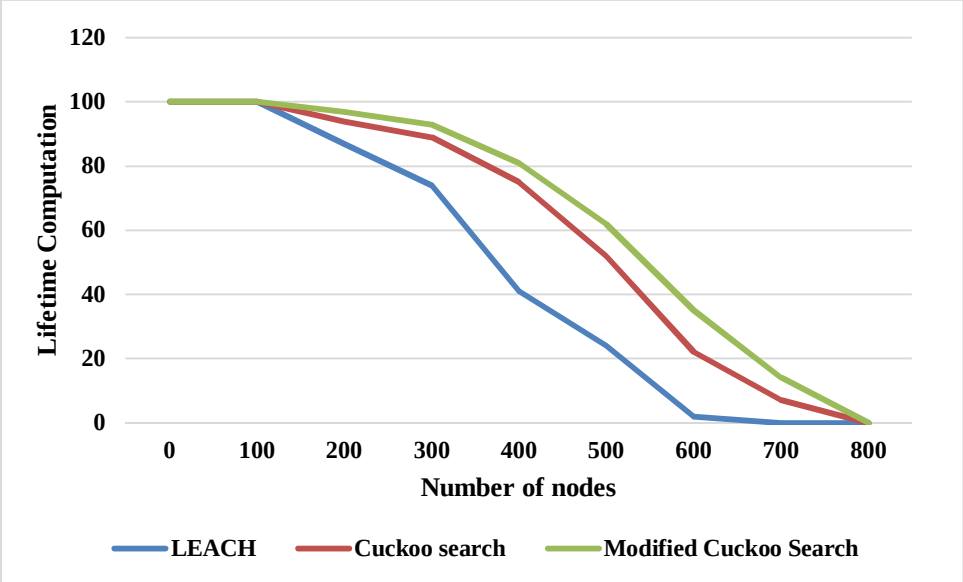


**Figure 5.4 Lifetime computation**

The modified CS reduced packet loss rate averagely by 26.9481% when compared with LEACH and averagely reduced by 2.7144% when compared with CS. The modified CS improved lifetime from 10.8696% to 200% when compared with LEACH. The CS improved lifetime of network from 7.7348% to 166.6667% when compared with LEACH.

## 5.5 CONCLUSION

The main thing is that it could be impossible to recharge the battery because the nodes are deployed and spread randomly in a hostile environment or any other area of interest such as unapproachable areas or the disaster locations for getting the required information. It is also possible to use the energy from the external environment e.g. using the solar cells as a power source. So it is clear that it is the energy is a main issue for the systems grounded on WSNs. So in this paper we used modified cuckoo search for the energy efficiency of the network. The experiments conducted with LEACH, CS and modified CS for number of clusters, end to end delay, packet loss rate and lifetime are calculated. The results proved that the modified CS outperformed than LEACH and CS.