Ain Shams Engineering Journal (2016) xxx, xxx–xxx

Ain Shams University

Ain Shams Engineering Journal

www.elsevier.com/locate/asej
www.sciencedirect.com

ELECTRICAL ENGINEERING

Lifetime Improvement in Wireless Sensor Networks using Hybrid Differential Evolution and Simulated Annealing (DESA)

Sweta Potthuri, T. Shankar *, **A. Rajesh**

School of Electronics Engineering, VIT University, Vellore, India

Received 11 August 2015; revised 15 February 2016; accepted 6 March 2016

KEYWORDS

Wireless Sensor Networks;
Differential Evolution;
LEACH;
Harmony Search;
Modified Harmony Search;
DESA

Abstract The major concerns in Wireless Sensor Networks (WSN) are energy efficiency as they utilize small sized batteries, which can neither be replaced nor be recharged. Hence, the energy must be optimally utilized in such battery operated networks. One of the traditional approaches to improve the energy efficiency is through clustering. In this paper, a hybrid differential evolution and simulated annealing (DESA) algorithm for clustering and choice of cluster heads is proposed. As cluster heads are usually overloaded with high number of sensor nodes, it tends to rapid death of nodes due to improper election of cluster heads. Hence, this paper aimed at prolonging the network lifetime of the network by preventing earlier death of cluster heads. The proposed DESA reduces the number of dead nodes than Low Energy Adaptive Clustering Hierarchy (LEACH) by 70%, Harmony Search Algorithm (HSA) by 50%, modified HSA by 40% and differential evolution by 60%.

1. Introduction

Wireless Sensor Networks (WSNs) consist of large number of tiny nodes which are either manually or randomly deployed in diversified applications. The network considered is of homogeneous or heterogeneous in nature. Eventually due to the energy consumption being dependent on the distance between the cluster head and the node or the base station and the node, the energy of the nodes reduces in each and every round. The network totally dies when all the nodes present in the network have zero energy. Thus, the energy must be optimally used [1]. Optimization is the procedure of finding the conditions that minimize the death of the nodes [2]. Optimization in general can be either heuristic or meta-heuristic. Heuristics work on one problem at a time. Meta-heuristics on the other hand work on a set of problems at a time. Heuristics take full advantage of the particularities of the problem and they are greedy in nature in the sense that the solution gets trapped in local minimum and fails to find the global optimum [3]. However, meta-heuristics are problem dependent and do not take advantage of the particularities of the problem and find

* Corresponding author. Tel.: +91 9486282025.
E-mail address: tshankar77@gmail.com (T. Shankar).

Peer review under responsibility of Ain Shams University.

ELSEVIER Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.asej.2016.03.004
2090-4479 © 2016 Faculty of Engineering, Ain Shams University. Production and hosting by Elsevier B.V.

the global minimum. On the other hand, heuristics is tailored to solve a specific problem.

Meta-heuristics are general algorithms and can be applied to various fields. But, heuristics uses a trial and error procedure to find acceptable solutions for complex problems in a reasonable amount of time. Meta heuristics might not give a best solution but gives a solution good enough or even optimal most of the time. Meta heuristics are incomplete methods and are black box procedures [29]. Meta Heuristics are embedded with operators to escape from local minimum. Unlike heuristic Dijkstra algorithm, Genetic algorithms, Differential Evolution (DE) [28], and Simulated Annealing (SA) are instances of meta-heuristics algorithm [29,30]. The method in [29] aimed to combine differential evolution with particle swarm optimization for node localization. The technique in [30] solves the cluster head selection using fuzzy concept. However, the fuzzy does not provide optimal solution but converges to the intermediate solution as it requires accurate and complete information about the network. Hence, this technique may not be appropriate for random deployment of sensor nodes.

Meta-heuristics approach has been utilized in wireless sensor networks for selection of cluster head or selection of a better set of population for the next generation in the case of evolutionary algorithms [3,31,32,35,36]. These algorithms make use of the fitness function to obtain a better offspring. In this paper, a hybrid algorithm using DE and SA is used for wireless sensor networks to improve the network lifetime. Alternatively, numerous works have been carried out in the area of network coding for cluster head selection [4] by considering the residual energy or the distance from base station [5,6].

The motivation of this paper is that during cluster formation, if the sensor nodes are not assigned properly to the cluster heads then the cluster heads eventually get over loaded, which leads to earlier death of the cluster heads [24,25]. This increases the latency and reduces the performance of the network. In this paper, we make use of DE for local search along with SA for global optimal solutions. Here, the proposed hybrid differential evolution and simulated annealing (DESA) algorithm aim at maximizing the network lifetime of the WSN by optimal search of the cluster heads. The results show that the proposed DESA outperforms the existing Low Energy Adaptive Clustering Hierarchy (LEACH), differential evolution, Harmony Search Algorithm (HSA) and Modified Harmony Search Algorithm (MHSA).

The rest of the paper is organized as follows. Section 2 describes the related work for lifetime improvement in wireless sensor network. In Section 3, the system model of the wireless sensor network has been detailed. Section 4 details the proposed differential evolution and simulated annealing algorithm. In Section 5, simulation results are discussed and Section 6 concludes the major findings of the paper.

2. Related works

There are number of clustering protocols proposed in the literature [7]. Low Energy Adaptive Clustering Hierarchy (LEACH) has been the most well-known algorithms that forms nodes based on received signal strength and random probabilistic distribution. LEACH is used in WSN for minimization of energy dissipation [9]. It is a cluster based protocol which elects cluster heads on a round basis. The protocol works in two phases namely set up and steady state phase. It starts with the set up phase where the cluster heads are chosen followed by the steady state phase where the data are transmitted. LEACH is a stochastic protocol which makes use of a probability p. Here, r represents the round, G is the set of all the nodes that are eligible to become cluster heads and p is the probability that each of the nodes will become the cluster heads. For each sensor node a random number in the range $[0,1]$ is chosen. If the number is lesser than the threshold, T, the node is chosen as the cluster head.

$$T_i(t) = \begin{cases} \frac{p}{1-p*\left(r \bmod \frac{1}{p}\right)} & ; \quad n \in G \\ 0 & ; \quad n \notin G \end{cases} \tag{1}$$

In addition to LEACH, Particle Swarm Optimization (PSO) and ant colony optimization (ACO) are widely evolutionary based approaches used in WSNs [3,10–13,16]. However, the abovementioned evolutionary clustering algorithms form the clusters by modest CH selection and allowing the non-CHs to join their nearest CHs. Moreover, they assume that the sensor nodes are equally distributed. Therefore, if the non-CH sensor nodes join the nearest CH as with LEACH then the CHs of densely deployed areas will be overloaded with higher number of member sensor nodes.

In [24,25], the authors have exploited PSO for clustering and routing in wireless sensor networks. Though PSO is a very efficient optimizer it suffers from curse of dimensionality [26]. However, the performance of Differential Evolution (DE) has been proved to be outstanding in comparison with the other conventional algorithms for clustering in WSN [27]. It is simple and robust, converges fast, and finds the optimum in almost every run. In addition, it has few parameters to set, and the same settings can be used for many different problems. Among the various algorithms, the DE can rightfully be regarded as an excellent candidate, when faced with new optimization problems [27,28].

The Harmony Search Algorithm (HSA) and Modified HSA (MHSA) exploited in [19,20], respectively, suffer from fixed pitch adjustment rate that sources uncertain and random search directions. This uncertainty causes them to obtain local optimal solution and the random search reasons slow convergence toward optimum value. Hence, it is very time consuming for the improvization of cluster head selection and during the searching of new cluster head.

The DE [21–23] advantages the above limitation by enhancing the capacity of local search by keeping the multiplicity of the population. The reason is that the DE does not require much fine tuning of the cross over rate parameter as the crossover operator shuffles information about positive combinations. Conversely, the DE does not guarantee an optimal solution as the convergence is unstable and more often locks itself into the regional optimal solution. Unlike binary Particle Swarm Optimization (PSO) [29], the DE supports only real number based decision variables.

The SA [17,18] mechanism advantages from finding the global optimal solution as it uses probabilistic jump during local optimal solution and avoids search process in local minimum. However, the simulated annealing cannot guarantee the finding of optimal solution. Hence, it takes longer period for optimization process that results in lower convergence speed. In addition, the simulated annealing tends to provide many solutions and few of them are not optimal.

2.1. Differential evolution

Differential evolution is a meta-heuristic evolutionary algorithm [8]. It is a robust algorithm, which aims at global optimization. Also, DE is a simple, easily adaptable algorithm for optimization of multimodal search spaces. DE is similar to Genetic Algorithm in the sense that both of them are evolutionary approaches. There exist some changes from GA though. GA applies crossover first followed by mutation which is the other way round in the case of DE. In GA, mutation is used to maintain population diversity and hence applied occasionally. DE on the other hand uses mutation operation in every generation to produce a better offspring. GA usually makes use of binary representation while DE can make use of set of real numbers [21,22].

DE consists of the following steps and the flowchart is shown in Fig. 1.

- Initialization of population vector i is done using a random set of values with n being the maximum size of the population for a particular generation '*Gen*' and is given as follows:
$$X_{i,Gen} = [x_{1,Gen}, x_{2,Gen}, x_{3,Gen} \ldots, x_{n,Gen}] \quad (2)$$
- Mutation is the process of generating the donor vector using the target vectors.
$$v_{i,Gen+1} = x_{r1,Gen} + F(x_{r2,Gen} - x_{r3,Gen}) \quad (3)$$
where $r_{1,Gen}$, $r_{2,Gen}$, $r_{3,Gen}$ are three random values chosen which are in the range $\{1,2, \ldots n\}$ and F is the amplification factor of the differential variation.
- Crossover is the operation carried out after the mutation phase. The trial vectors are obtained in the crossover stage. A random number in the range $[0, 1]$ is compared with the crossover rate (Cr). If the crossover rate is lower than the random number, the donor vector is chosen else the same vector remains.
$$u_{i,Gen+1} = \begin{cases} v_{i,Gen+1}, & rand(i) < Cr \\ x_{i,Gen}, & rand(i) > Cr \end{cases} \quad (4)$$
- Selection is an important phase of choosing the vectors for the next generation. Either the trial vector or the target vector is chosen as the offspring.
$$x_{i,Gen+1} = \begin{cases} u_{i,Gen}, & fitness(u_{i,Gen}) \geqslant fitness(x_{i,Gen}) \\ x_{i,Gen}, & otherwise \end{cases} \quad (5)$$

2.2. Simulated annealing

Simulated Annealing is a meta-heuristic algorithm used in materials which uses the concept of not ruling out the worse solution. This is very useful as the solution might not be the worst during the first little iteration [14]. Although the solution does not satisfy the criterion, the solution is not rejected straight away but is rejected with a probability as follows:

$$p = \exp\left(-\frac{\Delta E}{k_B T}\right) \quad (6)$$

The change in energy in Eq. (8) is represented in Eq. (9) as follows:

$$\Delta E = \gamma \Delta f \quad (7)$$

where Δf is the change in the fitness function and γ is chosen as the inverse of the Boltzmann's constant. On substituting change in energy in Eq. (8), the resultant probability is given as follows:

$$p = \exp\left(-\frac{\Delta f}{T}\right) \quad (8)$$

where ΔE is the change in energy levels, k_B is the Boltzmann's constant, T is the temperature for controlling the annealing process, which is chosen as the average value of the fitness function and γ is chosen as the inverse of the Boltzmann's constant. Thus, the probability is expressed as an exponential function of the difference in fitness function and the temperature.

3. System model

The radio dissipation model is a free space model which consists of the transmitter and receiver section with a separation of distance, d. The transmission section consists of transmit electronics and transmission amplifier and the receiving section consists of receive electronics part for information to be transmitted in terms of bits. Assume a set of sensors is dispersed on a rectangular field [9]. The energy required by the sensor nodes to transmit (E_{Tx}) and receive (E_{Rx}) the information (k) over the distance d given in Eqs. (9) and (10), respectively, is as follows:

$$E_{Tx} = \begin{cases} kE_{elec} + k\varepsilon_{fs}d^2, & d \leqslant d_0 \\ kE_{elec} + k\varepsilon_{mp}d^4, & d > d_0 \end{cases} \quad (9)$$

$$E_{Rx} = kE_{elec} \quad (10)$$

E_{elec} is the energy consumed to send one bit of data; ε_{fs} is the amplification coefficient of the transmission amplifier in free space, and ε_{mp} is the coefficient of amplifier under multipath consideration. The following properties about the network are anticipated:

- The nodes in the network are considered to be quasi-stationary in nature.
- The energy consumption is not uniform for all the nodes and depends on the distance from the base station or the cluster head depending on whichever is closer.
- Nodes are unaware of the location.
- All the nodes are homogeneous in nature.
- The nodes are self-organizing and need not be monitored after deployment.
- Each node has a fixed number of power levels.

4. Proposed Differential Evolution and Simulated Annealing (DESA) algorithm

The Differential evolution and simulated annealing (DESA) algorithm consists of the following steps:

- The set of sensor nodes is denoted by $S = \{s_1, s_2, \ldots, s_n\}$.
- From the set of sensor nodes we choose a small percentage of them as cluster heads.

Please cite this article in press as: Potthuri S et al., Lifetime Improvement in Wireless Sensor Networks using Hybrid Differential Evolution and Simulated Annealing (DESA), Ain Shams Eng J (2016), http://dx.doi.org/10.1016/j.asej.2016.03.004

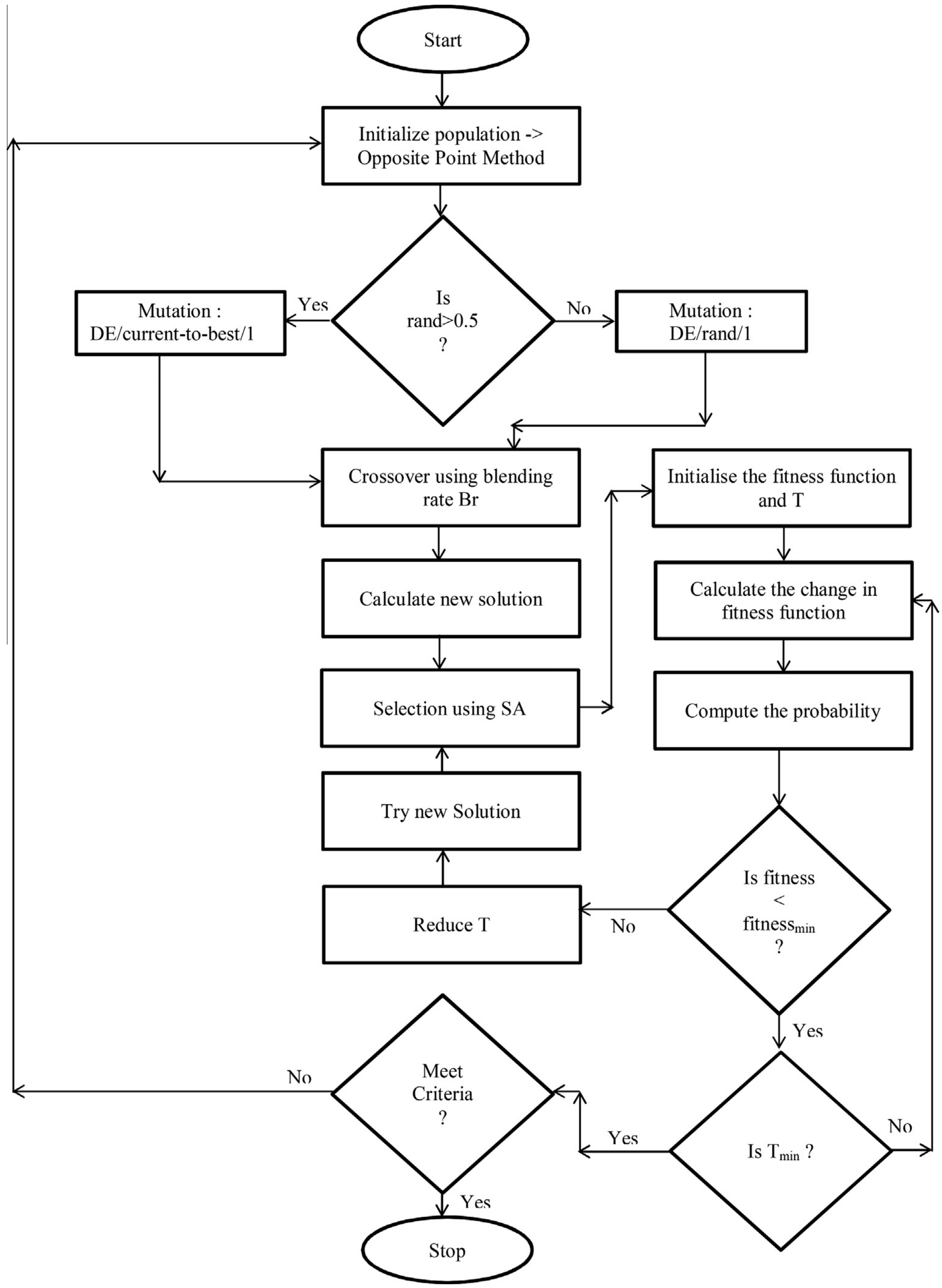

Figure 1 Proposed Differential Evolution and Simulated Annealing (DESA) algorithm.

- All the nodes are assigned to cluster heads using the procedure given in Section 4.7.
- Now, we go ahead with initialization of the population vector followed by mutation, crossover and selection using the simulated annealing meta-heuristic algorithm.

The DESA consists of four phases, namely initialization of the population vector, mutation, crossover and selection for the next generation as carried out in the conventional DE algorithm. In this paper, initialization of the population vector is done using the opposite point method. Here,

- The population is first initialized randomly.
- The value is optimized by generation another set of population vectors from the initial set called the opposite population.
- From the entire set, only ‘n’ fittest individuals are considered for the following generation.

For Mutation a random number in the range $[0,1]$ is first chosen. In the proposed hybrid approach, a threshold of 0.5 is assumed to choose between various DE schemes. In the literature [15] and the references therein, the notation of DE/x/y

is used, which is similar to Kendall's notation in queuing theory. Here, x represents mutation vector that is randomly chosen, y is the number of variance in vectors used in the mutation process and z refers to the crossover scheme, which usually takes the value of binomial or exponential. In the proposed approach, if the random number chosen is greater than a threshold value it performs DE/rand/1 else it performs DE/current-to-best/1. The z factor for crossover scheme is chosen as in Eq. (14).

For crossover we used the blending rate using the Gaussian distribution. In the selection phase, the fittest offspring is chosen for the next generation using selection using simulated annealing. Differential evolution depends on three control parameters namely population size, amplification factor and the crossover rate. The amplification factor and the crossover rate are made self-adaptive. The hybrid algorithm is depicted by the flowchart in Fig. 1. Each of the steps in Fig. 1 is discussed in detail in the following sections.

4.1. Initialization of population vectors using the opposite point method

The opposite point method is more effective than just the randomly chosen population for global optimization problems. Evaluating the opposite point simultaneously provides us another opportunity to find a point closer to the global optimum. The opposite point method must be applied before and after each round [23]. Here, the aim is to find the best set of population. It involves the following:

- Initialize the population randomly at the start as with the size population.
- Calculate the opposite population using the opposite point method.
- Find the union of the set of the population randomly selected and the opposite population.
- From the union, select the n fittest individuals. The same procedure needs to be repeated after the selection of the offspring for the next round.
- The next generation is obtained after the DE operations, namely, mutation, crossover, and selection.
- Obtain the opposite population.
- Select the n fittest individuals.
- Increment the next generation.

4.2. Self adaptive control parameters

The control parameters for Differential Evolution are amplification factor, crossover rate and the population size. In the proposed method, the Amplification Factor (F) and the crossover rate (CR) are made self-adaptive hence change in every round and for every individual to get better results [18].

$$F_{i,Gen+1} = \begin{cases} F_l + rand_1 * F_u \ (rand_2 < \tau_1) \\ F_{i,Gen} \quad otherwise \end{cases} \tag{11}$$

$$CR_{i,Gen+1} = \begin{cases} rand_3 \ (rand_4 < \tau_2) \\ CR_{i,Gen} \quad otherwise \end{cases} \tag{12}$$

Here, $rand_j, j \in \{1,2,3,4\}$ are uniform random values and τ_1 and τ_2 represent probabilities to adjust factors F and CR respectively. Upper bound (F_u) is chosen as 0.9 and lower bound (F_l) is chosen as 0.1. The probabilities τ_1 and τ_2 are chosen as 0.1 to adjust the amplification factor. They influence the mutation, crossover, and selection operations of the new vector $x_{i,G+1}$.

4.3. Mutation

A random number in the range [0, 1] is chosen and compared with a threshold value, in this case 0.5. If the random number chosen is lesser than the threshold DE/rand/1 is chosen else DE/current-to-best/1 is performed.

$$v_{i,Gen+1} = \begin{cases} x_{r1,Gen} + F_{i,Gen+1}(x_{r2,Gen} - x_{r3,Gen}) & rand[0,1] \leqslant 0.5 \\ x_{i,Gen} + F_{i,Gen+1}(x_{best,Gen} - x_{i,Gen}) & \\ +F_{i,Gen+1}(x_{r2,Gen} - x_{r3,Gen}) & otherwise \end{cases} \tag{13}$$

4.4. Crossover

A part of the mutant vectors is kept intact as in classical DE with probability Cr. The other features are not directly taken from the parent vector but are considered to be a mix, in a definite ratio. The blending rate Br determines the rate by which the mix occurs [19]. As we can observe if the value of the blending rate, Br is taken to be zero then the $u_{j,i,Gen}$ vector becomes $v_{j,i,Gen}$ for the next generation.

$$u_{j,i,G} = \mathrm{Br} * x_{j,i} + (1 - Br) * v_{j,i,G} \tag{14}$$

where the blending rate (Br) is $\mathrm{Br} = N(0.5, (1/2\pi))$ and $N(\mu, \sigma)$ is the normal distribution with mean μ and standard deviation σ.

4.5. Selection

$$x_{i,G+1} = \begin{cases} u_{i,G} & fitness(u_{i,G}) \geqslant fitness(x_{i,G}) \\ \begin{cases} u_{i,G} & choose\ with\ a\ probabilty\ of\ p_i \\ x_{i,G} & otherwise \end{cases} \end{cases} \tag{15}$$

Depending on the fitness function either u or x is chosen for the next generation [20]. If the fitness function of solution after crossover operation is greater than that of the original solution u is chosen else x is chosen with a probability. The probability is expressed in exponential form of the difference of the fitness function and the average fitness value in each round. The idea behind using the probability is that the worse solution at each round is not rejected straight away but depends on the probability given by

$$probability = \exp\left(-\frac{(fitness_{u_i(t+1)} - fitness_{x_i(t)})}{average(fitness)}\right) \tag{16}$$

4.6. Fitness function

The aim of using the fitness function is to get the best set of population vectors for the next generation. Distance and the

Please cite this article in press as: Potthuri S et al., Lifetime Improvement in Wireless Sensor Networks using Hybrid Differential Evolution and Simulated Annealing (DESA), Ain Shams Eng J (2016), http://dx.doi.org/10.1016/j.asej.2016.03.004

energies of the nodes are the factors taken into account as follows [21]:

$$fitness = \varepsilon * f_1 + (1 - \varepsilon) * f_2 \tag{17}$$

$$f_1(i) = \frac{E(i)}{\sum_{k=1,k \neq i}^{m} E(k)} \tag{18}$$

$$f_2(i) = \frac{(m-1)}{\sum_{k=1,k \neq i}^{m} d(i,k)} \tag{19}$$

where ε is an user defined constant, it determines the contribution of each of the functions used, f_1 is the ratio of the energy of the present node to the energy of the nodes in the cluster and f_2 is the total Euclidean distance of the cluster nodes to node i and $d(i,k)$ refers to the distance between node i and node k and m is the number of nodes in the particular cluster.

4.7. Cluster head distribution

After obtaining the set of the cluster heads and the set of the sensor nodes, the task of allotting a cluster head to each of the sensor nodes is carried out using the following procedure:

- The set of cluster head, ComCH (s_j) is obtained by the set of nodes which are within the radius of 20 m from the sensor node.
- The index number of ComCH is considered.
- The population $x_{i,G}$, is a random number generated in the range [0, 1].
- The ceil of ($x_{i,G}$ * ComCH(s_j)) is computed.
- The cluster head is assigned using the obtained number. If the ceil value is found to be 2, the next value of the set ComCH(s_j) is selected.

The resultant cluster head selection using the proposed method is shown in Table 1.

5. Simulation results and discussion

The simulation was carried out using Matlab 2015a simulator. Along with DESA, the performance of the network is analyzed for LEACH, HSA, modified HSA and differential evolution algorithm. The parameters [33,34] taken into consideration for simulating the network are shown in Table 2. The network parameters such as the number of dead nodes, number of alive nodes, energy consumption and throughput are analyzed and plotted against the number of rounds. The wireless sensor network consists of nodes deployed randomly as shown in Fig. 2.

Table 2 Simulation parameters.

Parameter	Value
Area	$200 * 200\ m^2$
No. of nodes	100
Initial energy of nodes	0.5 J
k, Packet size	4000
E_{elec}	70 nJ
E_{DA}	5 nJ
E_{fs}	10 pJ
E_{mp}	0.0013 pJ
E_{amp}	120 nJ

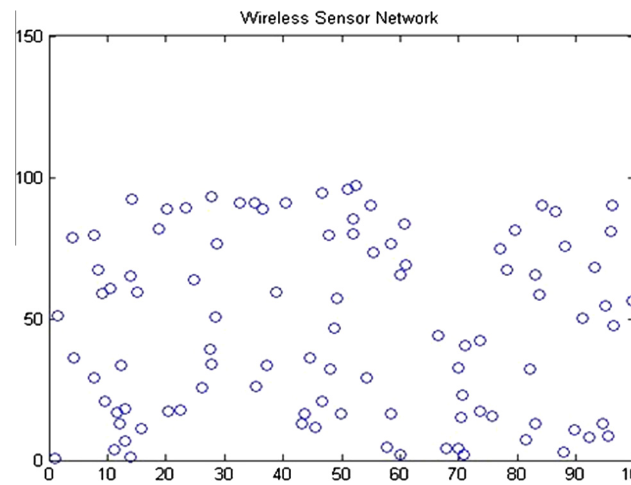

Figure 2 Random deployment of wireless sensor network.

These are now grouped into small sectors called clusters. The results have been obtained after averaging the results from 15 seeds and each seed is carried out for 1000 runs.

The number of dead nodes with varying number of rounds is shown in Fig. 3. The number of dead nodes must be as minimal as possible for better performance of the network. If the cluster heads die earlier, the clusters will be left without any CH. Such scenarios should be avoided. From the results obtained we observe that using DESA there is an increase of 70% when compared to LEACH, 50% when compared to HAS, 40% when compared to modified HSA and 60% when compared to differential evolution, respectively. The nodes are alive for longer duration due to the hybridization of differential evolution algorithm along with the modifications simulated annealing approach. Hence, the hybrid method provides better results as compared to other algorithms.

Table 1 Formation of cluster head in the proposed algorithm.

Sensor	ComCH(s_j)	|ComCH(s_j)|	$x_{i,Gen}$	Ceil($x_{i,Gen}$ * ComCH(s_j))	Assigned CH
s1	{CH3, CH1, CH2}	3	0.46	2	CH1
s2	{CH4, CH5}	2	0.19	1	CH4
s3	{CH5,CH3}	2	0.39	1	CH3
s4	{CH1,CH4, CH5,CH3}	4	0.67	3	CH5
s5	{CH5}	1	0.86	1	CH5
s6	{CH1,CH2}	2	0.63	2	CH2
s7	{CH2}	1	0.24	1	CH2

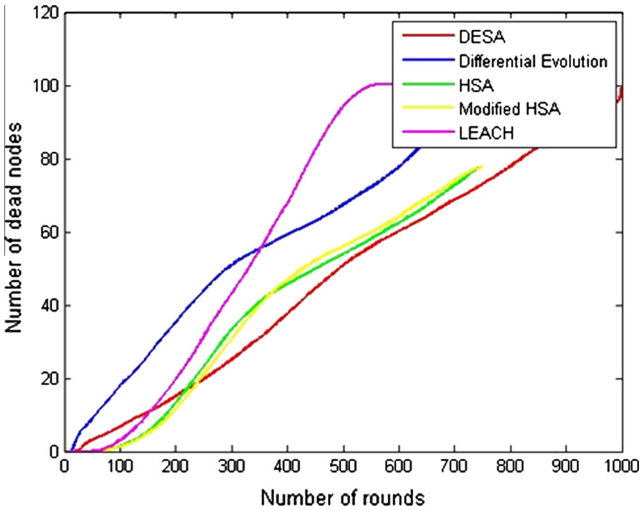

Figure 3 Number of dead nodes with varying number of rounds.

The number of alive nodes for varying number of nodes is shown in Fig. 4. From the curves it could be observed that the number of alive node with DESA is increased than that of LEACH, HSA, modified HSA and differential evolution algorithms. This proves that the nodes stay alive in the network for longer period and reduce the probability of early death of the nodes. This is due to the fact that the cluster heads are assigned to the nodes by the method discussed in cluster head allocation of DESA and much of the energy is not being consumed.

The residual energy of the network with varying number of rounds is shown in Fig. 5. It is the energy left after each and every round. The batteries used in the wireless sensor networks are very small in size and cannot be replaced; hence, the energy consumption by residual energy must be as minimal as possible. From the curves we could infer that in the proposed DESA algorithm, the residual energy increases as the nodes die gradually. This is due to the suitable choice of fitness function chosen, which considers the residual energy and the distance. Fig. 6 shows the throughput with varying number of rounds. Throughput of the network indicates the amount of data being sent in each round. As the number of alive nodes increases the throughput of the network also increases. Throughput of the network is calculated as the product of the number of alive nodes per round and the data packet length. From the curves, the proposed hydride DESA algorithm outperforms LEACH, MHSA, HAS and DE algorithms.

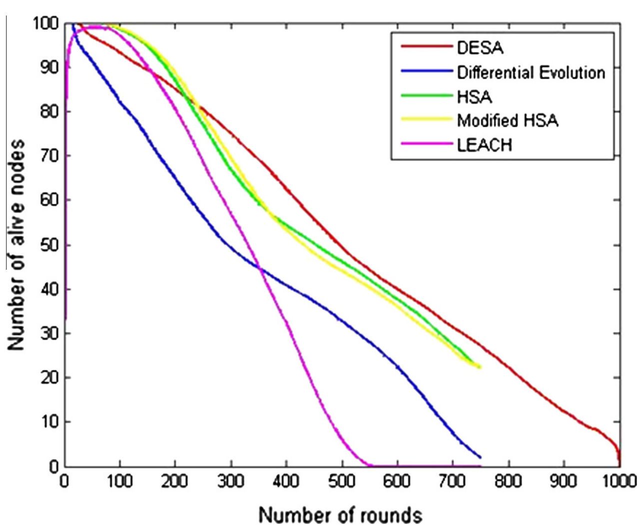

Figure 4 Number of alive nodes with varying number of rounds.

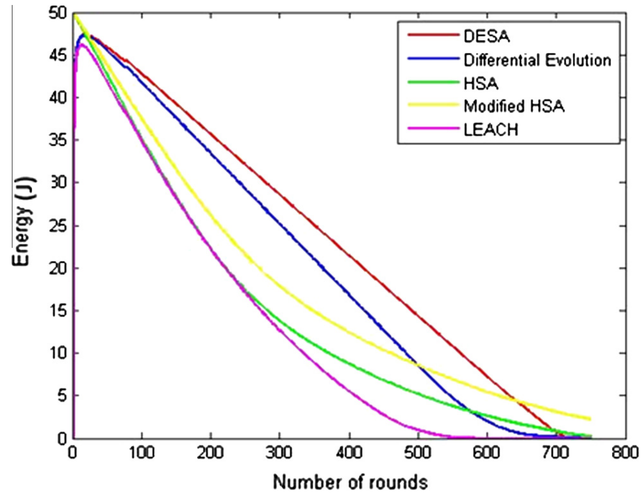

Figure 5 Residual energy with varying number of rounds.

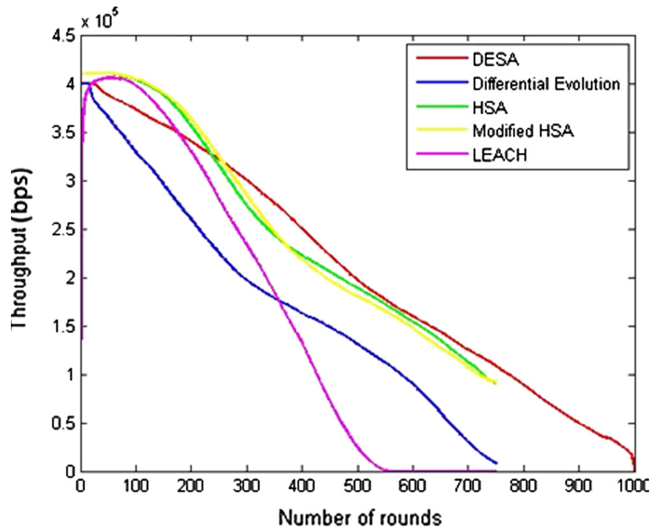

Figure 6 Throughput with varying number of rounds.

6. Conclusion

In this paper, a hybrid Differential Evolution and Simulated Annealing (DESA) was proposed, which is a hybrid of Differential Evolution and Simulated Annealing. It is used to improve the network lifetime by prolonging the death of the cluster heads. DESA includes a fitness function taking into consideration the residual energy and distance between the cluster head and the nodes. Among the various methods, the experimental results have shown that the network lifetime with DESA algorithm has been improved by 40% as compared to modified HSA algorithm.

References

[1] Akyildiz IF, Su W, Sankarasubramaniam Y, Cayirci E. Survey on sensor Networks. Commun Mag IEEE 2002;40:102–14.

[2] Shankar T, Shanmugavel S, Karthikeryan A. Hybrid approach for energy optimization in wireless sensor networks using PSO. Int Rev Comput Softw 2013;8(6):1454–9.

[3] Yang XS. Nature-inspired metaheuristic algorithms. Univer Press; 2008.

[4] Muruganathan SD, Ma DCF, Bhasin RI, Fapojuwo A. A centralized energy-efficient routing protocol for wireless sensor networks. IEEE Commun Mag 2005;43(3):8–13.

[5] Shankar T, Shanmugavel S, Karthikeyan A, Akanksha Mohan Gupte, Suryalok. Load balancing and optimization of network lifetime by use of double cluster head clustering algorithm and its comparison with various extended LEACH versions. Int Rev Comput Softw 2013;8(3):795–803.

[6] Heinzelman W, Chandrakasan A, Balakrishnan H. Application specific architecture for wireless micro sensor networks. Wireless Commun IEEE 2000;1:660–70.

[7] Shankar T, Shanmugavel S. Energy optimization in cluster based wireless sensor networks. J Eng Sci Technol 2014;9(2):246–60.

[8] Storn Rainer, Price Kenneth. Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 1997:341–59.

[9] Wang A, Heizelman W, Chandrakasan A. Energy-scalable protocols for battery operated microsensor networks. In: Proceeding IEEE workshop signal processing systems (SiPS). p. 483–92.

[10] Zungeru AM, Ang LM, Seng KP. Classical and swarm intelligence based routing protocols for wireless sensor networks: a survey and comparison. J Netw Comput Appl 2012;35(5):1508–36.

[11] Saleem M, Gianni A, Di Caro MF. Swarm intelligence based routing protocol for wireless sensor networks: survey and future directions. Inform Sci 2011;181(20):4597–624.

[12] Kulkarni RV, Venayagamoorthy GK. Particle swarm optimization in wireless-sensor networks: a brief survey. IEEE Trans Syst Man Cybern C: Appl Rev 2011;41(2):262–7.

[13] Mukherjee R, Debchoudhury S, Kundu R, Das S, Suganthan P. Adaptive differential evolution with locality based crossover for dynamic optimization. In: Evolutionary Computation (CEC) IEEE Congress. p. 63–70.

[14] Das S, Konar A, Chakraborty UK. Annealed differential evolution. In: Evolutionary Computation. IEEE Congress on CEC. p. 1926–33.

[15] Kuila Pratyay, Prasanta KJ. A novel differential evolution based clustering algorithm for wireless sensor networks. Appl Soft Comput 2014;25:414–25.

[16] Pratyay K, Gupta SK, Jana PK. A novel evolutionary approach for load balanced clustering problem for wireless sensor networks. Swarm Evol Comput 2013;12:48–56.

[17] Kirkpatrick S, Gelatt Jr, Vecchi MP. Optimization by simulated annealing. Science 1983;220(4598):671–80.

[18] Shankar T, Shanmugavel S. Hybrid approach for energy optimization in wireless sensor network using ABC and firefly. Int Rev Comput Softw 2013;8(10):2335–41.

[19] Hoang DC, Yadav P, Kumar R, Panda SK. A robust harmony search algorithm based clustering protocol for wireless sensor networks. In: Communications Workshops (ICC) IEEE International Conference. p. 1–5.

[20] Geem ZW, Kim JH, Loganathan GV. A new heuristic optimization algorithm: harmony search. Simulation 2001;76(2):60–8.

[21] Kuila P, Jana PK. A novel differential evolution based clustering algorithm for wireless sensor networks. Appl Soft Comput 2014;25:414–25.

[22] Youyun A, Hongqin Chi. Differential evolution using opposite point for global numerical optimization. J Intell Learn Syst Appl 2012;4:1–19.

[23] Janez Brest, Saso Greiner, Borko Bo skovic, Marjan Mernik, Viljem Zumer. Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. IEEE Trans Evol Comput 2006;10(6):646–57.

[24] Kuila P, Jana PK. Energy efficient clustering and routing algorithms for wireless sensor networks: particle swarm optimization approach. Eng Appl Artif Intell 2014;33:127–40.

[25] Kim D, Song S, Choi BY. Energy-efficient adaptive geo source multicast routing for wireless sensor networks. J Sens 2013;14.

[26] Jasper J, Victoire TAA. Dispatching a 19-unit indian utility system using a refined differential evolution algorithm. Math Prob Eng 2014;12.

[27] Vesterstrom J, Thomsen R. A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems. Proc Congr Evol Comput 2004;2:1980–7.

[28] Storn R, Price K. Differential evolution a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 1997;11(4):341–59.

[29] Lieping Zhang, Wenjun Ji, Yu Zhang. Node localization method for wireless sensor networks based on hybrid optimization of differential evolution and particle swarm algorithm. Open Autom Contr Syst J 2014;6:621–8, January 19, 2015.

[30] Puneet Azad and Vidushi Sharma, Cluster Head Selection in Wireless Sensor Networks under Fuzzy Environment, ISRN Sensor Networks, Volume 2013 (2013), Article ID 909086, 8 pages.

[31] Valipour M, Montazar AA. Sensitive analysis of optimized infiltration parameters in SWDC model. Adv Environ Biol 2012;6(9):2574–81.

[32] Valipour M, Montazar AA. Optimize of all effective infiltration parameters in furrow irrigation using visual basic and genetic algorithm programming. Aust J Basic Appl Sci 2012;6(6):132–7.

[33] Amgoth Tarachand, Jana Prasanta K. Energy-aware routing algorithm for wireless sensor networks. Elsevier J Comput Electr Eng 2015;41:357–67.

[34] Wendi Rabiner Heinzelman, Anantha Chandrakasan, Hari Balakrishnan. Energy-efficient communication protocols for wireless micro-sensor networks. In: Hawaii International Conference on Systems Sciences.

[35] Valipour M, Banihabib ME, Behbahani SMR. Monthly inflow forecasting using autoregressive artificial neural network. J Appl Sci 2012;12(20):2139–47.

[36] Valipour Mohammad, Mohammad Ebrahim Banihabib, Seyyed Mahmood Reza Behbahani. Comparison of the ARMA, ARIMA, and the autoregressive artificial neural network models in forecasting the monthly inflow of Dez dam reservoir. J Hydrol 2013;476:433–41.

Please cite this article in press as: Potthuri S et al., Lifetime Improvement in Wireless Sensor Networks using Hybrid Differential Evolution and Simulated Annealing (DESA), Ain Shams Eng J (2016), http://dx.doi.org/10.1016/j.asej.2016.03.004