Journal of Advance Computational Research
Vol. 3 No. 1 (January-June, 2018)

A Genetic Algorithm inspired Routing Protocol for Wireless Sensor Networks

Ayon Chakraborty[1], Swarup Kumar Mitra[2] and Mrinal Kanti Naskar[2]
[1]Department of Computer Science and Engineering, Jadavpur University, Kolkata, India
[2]Department of Electronics and Telecommunications Engineering, Jadavpur University, Kolkata, India
E-mails: jucse.ayon, swarup.subha@ gmail.com, mrinalnaskar @ yahoo.co.in

RECEIVED: December 04,2017. Revised March 08, 2018

Abstract: The key parameters that need to be addressed while designing protocols for sensor networks are its energy awareness and computational feasibility in resource constrained sensor nodes. Variation in the distances of nodes from the Base Station and differences in inter-nodal distances are primary factors causing unequal energy dissipation among the nodes. Thus energy difference between the various nodes increases with time resulting in degraded network performance. The LEACH and PEGASIS schemes which provided elegant solutions to the problem suffer basic drawbacks due to randomization of cluster heads and greedy chain formation respectively. In this paper, we propose a Genetic algorithm inspired ROUting Protocol (GROUP) which shows enhanced performance in terms of energy efficiency and network lifetime over other schemes. GROUP increases the network performance by ensuring a sub-optimal energy dissipation of the individual nodes despite their random deployment. It employs modern heuristics like Genetic Algorithms along with Simulated Annealing, instead of the greedy algorithm as in PEGASIS to construct energy efficient routing paths. Extensive simulations validate the improved performance of GROUP. GROUP is also tested for its computational feasibility in real sensor motes running under TinyOS software platform.

Keywords: Network Lifetime, Wireless sensor networks, Routing chain, Genetic Algorithm, Simulated Annealing

1. INTRODUCTION

Wireless sensor networks (WSNs) consist of resource constrained sensor nodes that are usually randomly deployed in a large area, collecting important information from the sensor field and transmitting the gathered data to a distant Base Station [1] [2]. It is often infeasible to replace or recharge the sensors nodes as they are deployed in inaccessible terrains. Thus energy efficiency in the sensor network protocols is an important issue to be addressed. Network lifetime thus becomes an important metric for efficiency of a sensor network protocol. In case of WSNs, the definition of network lifetime is application specific [3]. It may be taken as the time from inception to the time when the network becomes nonfunctional. A network may become non-functional when a single node dies or when a particular percentage of nodes perish depending on requirement. However, it is universally acknowledged that equal energy dissipation for equalizing the residual energy of the nodes is one of the keys for prolonging the lifetime of the network [2][3]. Thus design of energy efficient protocols is an important challenge particularly in the domain of data routing, which is one of the most important functions of the WSN.

Heinzelman *et al.* in [4] developed a cluster-based routing scheme called Low-Energy Adaptive Clustering Hierarchy (LEACH), where in each cluster, member nodes adopt a Time Division Multiple Access (TDMA) protocol to transmit their data packets to the cluster head. After receiving data packets from all its local members, a cluster head performs data aggregation and sends the final aggregated packet to the Base Station under the Carrier Sense Multiple Access (CSMA) protocol. To avoid cluster heads dying quickly, LEACH rotates the roles of cluster heads among all the sensor nodes. In doing so, the energy load is distributed evenly across the network and network lifetime (in unit of data collection round) becomes much longer than the static clustering mechanism. Compared with the Minimum Transmission Energy (MTE) routing scheme [5], where communication distance is the only criterion for selecting low-energy routes, LEACH utilizes a more accurate energy model and offers much better performance in terms of energy efficiency and network lifetime. The Power Efficient Gathering in Sensor

Correspondence email address: **jucse.ayon@gmail.com**

Information Systems (PEGASIS) scheme proposed in [6] is based on a greedy chain, which starts from the farthest node from the Base Station. By connecting the last node on the chain to its closest unvisited neighbor, PEGASIS greatly reduces the total communication distance and achieves much better energy and lifetime performance than LEACH for different network sizes and topologies. This problem was even approached by modern heuristic techniques like Ant colony Optimization [7], trying to optimize energy dissipation. [6] tried to form an optimized chain for data gathering. The PEGASIS scheme depends upon a greedy chain formation whereas the LEACH scheme randomizes the leader selection in the network. While the greedy chain can not always guarantee minimal energy consumption, the randomized leader selection does not take into account the node's capability in being the leader, in terms of its energy content and transmit distance.

In this paper, we attack the problem from a new viewpoint and propose a Genetic algorithm inspired ROUting Protocol (GROUP). In GROUP, a chain is formed, but instead of allowing all nodes to become the leader, to communicate with the base station the same number of times, the network lifetime is increased by allowing the individual nodes to transmit unequal number of times to the base station depending on their residual energy and location. Furthermore, instead of forming a greedy chain, which may not always ensure minimum energy dissipation, we make use of modern heuristic optimization techniques like Genetic Algorithms [8]. The efficiency of the Genetic Algorithm is further increased by applying Simulated Annealing [9]. This results in an enhanced network performance as balanced energy dissipation by the individual nodes is achieved in the network. The results obtained by applying GROUP shows encouraging improvements over PEGASIS, LEACH and ACO schemes. The GROUP algorithm was implemented in nesC [10] for the TinyOS software platform. This not only signifies the coding feasibility of our scheme, but also verifies it for running on real hardware platforms (embedded systems like MicaZ or Mica2 sensor motes). The packet reception ratios in these schemes has also been studied using the interference model offered by TOSSIM [11] environment.

The rest of the paper is organized as follows: Section 2 describes our system model and Section 3 gradually develops our protocol introducing all the basic concepts and backgrounds required. Our scheme is evaluated by results obtained from extensive simulations in Section 4. Finally, we conclude in Section 5.

2. THE SYSTEM MODEL AND PROBLEM FORMULATION

Our aim in this paper is to maximize network lifetime by minimizing the total energy usage of the individual sensor nodes by formation of an optimal data-gathering chain using heuristic techniques. For justification of the efficiency of our protocol and fair comparison with previous works [4] [5] [6] [7], we choose to follow the similar assumptions for the system modeling. We briefly discuss some important features of our model in the following points.

The Network Model

The foundation of GROUP relies on the realization of a powerful Base Station which is connected to an adequate source of energy supply. Some of the important features of the sensor network are:

- The Base Station is fixed and located far away from the sensor nodes.
- The sensor nodes are static, energy constrained and homogeneous with a uniform initial energy allocation.
- The nodes are equipped with power control capabilities and omni directional antenna to control the direction and magnitude of transmitted power.
- Each node senses its vicinity at a fixed rate and always has data to send to the Base Station in every data gathering round.
- The inter-nodal distances are smaller compared to distance between the nodes and the Base Station.

The two key elements considered in the design of GROUP are the sensor nodes and the Base Station. The sensor nodes are capable of operating in two modes: *The Sensing Mode* and *The Leader Mode*. In the Sensing Mode, the nodes perform sensing tasks and relay the sensed data to the Leader node through a multihop routing chain. In the Leader Mode a node gathers data from the other nodes in the chain performs the final data fusion tasks and sends them to the Base Station. The Base Station on the other hand performs some of the crucial tasks like formation of the data gathering chain and selection of the Leader Node.

The Radio Model

We considered the first order radio model for calculation of the energy dissipation for data communication operations like transmission and reception. This is one of the most widely accepted and used models in literature for sensor network simulations and theoretical analysis. The energy spent by a node in transmitting a *k*-bit packet to another node *d* meters away, is given by:

$$E_{TX}(k, d) = (\xi_{\text{elec}} + \xi_{\text{amp}} * d^{n}) * k \quad (1)$$

and that for receiving the packet is,

$$E_{RX}(k) = \xi_{\text{elec}} * k \quad (2)$$

Here ξ_{elec} (50nJ/bit) is the energy dissipated per bit to run the radio electronics and ξ_{amp} is the energy required by the transmit amplifier to maintain an acceptable signal to noise ratio (SNR) in order to transfer data messages reliably. *n* is called the path loss exponent, whose value enhances with increasing channel non-linearity (usually, $2.0 \le n \le 4.0$). In our approach, we have used both the free space (distance^2 power loss) and the multipath fading (distance^4 power loss) channel modes. It is also assumed that the channel is symmetric so that the energy spent in transmitting a packet from node *i* to *j* is the same as that from node *j* to *i* for any given value of SNR. For communication among sensor nodes we take $n = 2$, and that between the leader and Base Station, we take $n = 4$, in (1). Value of $\xi_{amp} = 10\text{pJ/bit/m}^2$ for $n = 2$ and 0.0013pJ/bit/m^4 for $n = 4$. Now for all practical purposes, we can assume that the computational energy is much less than the communicational energy and thus can be neglected.

Problem Formulation

A routing chain is an ordered sequence of all the nodes in the network forming a chain-like structure. A particular node in the chain is selected as the leader node to communicate with the Base Station. A data gathering round consists of a time interval within which all the nodes generate a packet of its own and transmit it. Every node in the network on receiving a packet from the previous node fuses it with its own data and relays it to the next node in the chain in the direction of the leader node. Our aim is to minimize the energy dissipation in the nodes, by the formation of an optimal data gathering chain. Considering N nodes in the network, the total energy expended in a typical data gathering round is the summation of the energy dissipated by the individual sensor nodes and the leader. Assuming a constant packet size of *k*,

$$E_{TOTAL} = \left\{ \sum_{i=1}^{N-1} (\xi_{elec} + \xi_{amp} * d_i^2) + \xi_{elec} \right\} + (\xi_{elec} + \xi_{amp} * D^4) * k \quad (3)$$

In (3), d_i denotes the euclidean distance between the $(i + 1)^{\text{th}}$ node and the i^{th} node in the data gathering chain. *D* is the euclidean distance between the sensor node acting as the leader and the Base Station. The values of ξ_{elec} and ξ_{amp} are stated earlier. Here we impose a threshold value on d_i as d_{TH}. This ensures reliable communication in between the nodes reducing unwanted noise and packet loss probability.

3. PROPOSED ALGORITHM

Genetic Algorithm is mainly a probabilistic search algorithm based on the principles and concept of natural selection and evolution. At each generation it maintains a population of individuals where each individual is a coded form of a possible solution of the problem at hand and is called a chromosome. Each chromosome is evaluated by a function known as the fitness function which is usually called the cost function or the objective function of the corresponding optimization problem. Next new population is generated from the present one through selection, crossover, repair and mutation operations. Purpose of selection mechanism is to select more fit individuals (parents) for crossover and mutation. A crossover causes the exchange of genetic materials between parents to form offspring, where as mutation incorporates new genetic materials in the offspring. Implementation of the above mentioned components for our proposed algorithm are as follows.

Genes and Chromosome

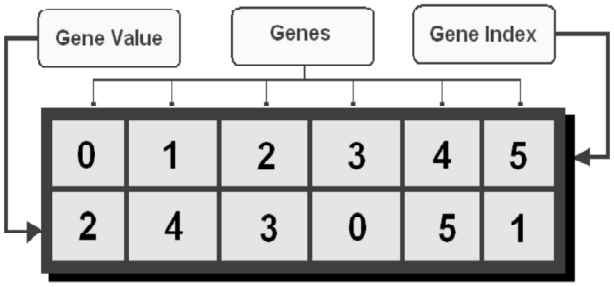

Figure 1: A Chromosome Containing 6 Genes, the Gene Indexes (Upper Row) and the Gene Values (Lower Row) are shown. Analogously, it Represents a Routing Chain Containing 6 Nodes

We represent our data routing chain by means of a chromosome, which contains genetic information for the Genetic Algorithm. A chromosome is a collection of genes and each chromosome represents a particular arrangement of the nodes in the routing chain for a given network. The *gene index* represents the position of the node in the chain and *gene value* provides the node's identification number (ID), as shown in the fig. 1. A chromosome is represented by an integer string of size N, where each integer is unique and lies between 1 and N. These integers are the individual gene values that make up the chromosome.

In this representation, each chromosome encodes an ordered sequence of nodes. Also note that for a chromosome containing N genes, there are N! different gene arrangements possible.

Initial Population

A population is a collection of chromosomes, where a family of *r* chromosomes is represented as P = (C_1, C_2 ... C_r). The population at the first generation has to be pre-generated to initiate the algorithm. The initial population has an effect over speed of convergence of an evolutionary algorithm depending on how much close the individuals (chromosomes) are toward the solution. In this paper we have generated the initial population with a simple random number generation algorithm. No efforts have been made to construct the initial population in some specific way as it is observed that the convergence of the solution is occuring within an acceptable number of generations. (eg., 50 generations can be considered as an acceptable limit of convergence for a genetic algorithm).

An initial population for GROUP is created with randomly chosen permutation of the integer string {1, 2… N}. It consists of *r* chromosomes, where *r* depends upon the extent of optimization. Greater the value of *r*, greater is the probability of reaching a better solution. Again, a greater *r* is more computation intensive, thus a trade off exists, depending on the context for the choice of *r*. While the chromosomes are generated, it is ensured that the distances between any two consecutive nodes or gene values, do not exceed the threshold distance for communication d_{TH}..

Algorithm 1 *Initial population generation algorithm*

inputs: **r** ← Number of individuals in a generation;
N ← Number of nodes in the WSN.

***output*:** $\mathbf{P^1}$ ← Initial Population

```
Procedure generate_initial_population() :
  number_of_chromosomes □! 0
    while number_of_chromosomes < r

    /* rand_string(a,b) creates a random string of the integers
lying between b and a, where a and b are integers and b>a */
  Ci = rand_string(1,N)

/* check(Ci , d_TH.) returns true if the i-th chromosome satisfies the
                    distance threshold, else false */
    if(check(Ci , d_TH.))
        increment number_of_chromosomes by 1
       add Ci to P1
        continue with the loop
    end if
  end while
end Procedure
```

Parent Selection

The selection process of Genetic Algorithm determines, each time, which two chromosomes out of the total population of *r* chromosomes will take part in mating to create offspring. We use Tournament Selection where two chromosomes are chosen randomly for mating.

Algorithm 2 *Parent selection algorithm*

inputs: $\mathbf{P^1}$ (initial population)
output: Two chromosomes C_{P1} and C_{P2} from the initial population $\mathbf{P^1}$

```
Procedure select_parents (P1) :
 do
    /* rand(1,N) generates a random number between 1 and N */
  i ← rand(1, N)
  j ← rand(1, N)
while i ≠ j
  C_P1 ← C_i
   C_P2 ← C_j
  end Procedure
```

Generation

A new generation is created using crossover and mutation operations, which are discussed in detail in the subsequent sections. The parent selection strategy stated in Algorithm 2, produces two chromosomes for crossover in order to produce new offspring. A single point crossover is performed between two parent chromosomes to produce an offspring. Moreover, the population size remains same for all the generations. We also keep a track of the historically obtained best chromosome, C_{best}.

Crossover: Crossover indicates the combination of the two parent chromosomes to produce an offspring.

We implemented the crossing method as discussed in [12]. For e.g., Say, C1 = {3,1,2,4,6,5} and C2 = {5, 2, 1, 4, 3, 6} are the two parent chromosomes selected for crossover. The slot {2, 1, 4} is randomly chosen from C2 and inserted in the same position in C1 and the node IDs that are repeated are deleted. Thus the offspring produced is O = {3, 2, 1, 4, 6, 5}. After the crossover, C1 is compared with C2 and the best one is considered for future mating.

Algorithm 3 *Crossover algorithm*

inputs: Two parent chromosomes, **C1 and C2**
output: Produced Offspring**, O**

Procedure crossover() :

do
/* N is the length of the chromosome */
gene_index_1 ← rand(1, N)
gene_index_2 ← rand(1, N)
while *gene_index_1* ≠ *gene_index_2*

gene_index_1 = min(*gene_index_1, gene_index_2*)
gene_index_2 = max(*gene_index_1, gene_index_2*)
slot ← sequence *of the* **gene values** *from* **gene_index_1** *to* **gene_index_2** *in* **C2**

Insert ***slot*** in **C1** starting from at the position *gene_index_1*
Delete ***repeated gene values*** from the remaining portion of **C1**
O ← C1

end Procedure

Evaluation and Fitness

In any evolutionary computation algorithm, a metric is needed to compare and evaluate the fittest candidate from a generation of candidates. The fitness of a candidate is determined by the objective of the optimization problem and the cost function is designed accordingly.

Simulated Annealing and Metropolis Acceptance: The basic idea of simulated annealing proposed by Metropolis was used to solve combinatorial optimization problems by Kirkpatrick. [9] It is a stochastic process that accepts the current optimal solution at a probability after searching, which is called the Metropolis Acceptance Law. The acceptance probability is determined by two factors, the *Energy Function*, which can be thought of to be similar to the fitness function of the Genetic Algorithm and the *Anneal Temperature*, θ. Starting from a higher temperature, θ is decreased to a lower limit, at a rate determined by a function called the *Cooling Schedule*. As θ approaches the lower-limit, searching region is constrained around the best point.

Fitness Function and Selection Methodology: Fitness is the core part of a Genetic Algorithm. Our fitness function is designed to increase the lifetime of the system, which evaluates whether a particular chromosome increases lifetime or not. We always preserve the historically obtained best chromosome, that is, with the highest fitness value. The Energy Function for the Simulated Annealing algorithm is designed as,

$$f(C) = \sum_{i=1}^{N} d_i^2 \tag{4}$$

The above equation is derived from (3); we have considered the terms in the chain related to distance only, which is responsible for energy consumption. Equation (4) calculates the *energy* of a chromosome C containing N genes and d_i denotes the euclidean distance between the $(i+1)^{th}$ node (or, gene) and the i^{th} node in the data gathering chain. It should be noted that a greater value of the chromosome energy indicates a longer chain and corresponds to an inferior solution. When Cparent1 and Cparent2, produces a new offspring Coffspring,

$$\Delta E = [f(\text{Coffspring}) - \{f(\text{Cparent1}) + f(\text{Cparent2})\} * 0.5\ ] \tag{5}$$

Equation (5) represents the energy difference between the two generations. Clearly, a negative ΔE indicates a superior offspring. But a negative ΔE does not always indicate the acceptance of the new offspring, which saves our algorithm from getting trapped in some local minimum. This is the context, where the Simulated Annealing algorithm comes into play, for the selection purpose. The acceptance of the new offspring is determined by the following Selection Algorithm, where the probability of an acceptance of a solution is determined by the technique of Simulated Annealing applied to combinatorial optimization as discussed in [9]. The pseudo code for the Selection Algorithm is presented below.

Algorithm 4 *Selection algorithm*

***inputs*:** Coffspring, Cparent 1, Cparent 2
output: Acceptance probability of Coffspring (either 0 or 1)

Procedure select():
/* offspring has lower energy state */
if *sign(ΔE) = -1*
/*accept new offspring */
acceptance_probability = 1
end if

```
    else /* sign (ΔE) = 1, offspring has higher energy state */
        temp = exp( – ΔE / θ)
         /* rand(0,1) returns a random number between 0 & 1 */
         if temp > rand(0,1)
          /* accept new offspring */
         acceptance_probability = 1
        end if
        else
          /* reject new offspring */
         acceptance_probability = 0
        end else
    end else
end Procedure
```

In the above algorithm, sign(x) returns 1, if x is non-negative, else return -1 and exp(x) denotes the exponentiation of x.

Repair and Mutation

Repair: If a produced offspring violates the constraints imposed in the optimization problem, namely the distance threshold, the solution is rejected, and the crossover operation is performed again. The Algorithm for the repair method is presented below.

Algorithm 5 *Repair algorithm*

```
Procedure repair(chromosome) :
  while acceptance_probability ≠ 1
     for every two successive genes i and i + 1 in the current
        offspring chromosome
       /* d_i = euclidean distance between i and (i +1)^th node */
       if d_i ≥ d_TH
       /* reject the offspring */
       acceptance_probability = 0
       break out of for loop
       end if
    end for
    if acceptance_probability = 0
      perform Crossover to produce new offspring /* Algorithm
                3 */
      Update current offspring chromosome
      acceptance_probability = 1
    end if
    else /* acceptance_probability = 1*/
        break out of while loop
    end else
  end while
end Procedure
```

Mutation: The *mutation* adds variation in the next generation. In mutation, a node is randomly picked slot from the historically obtained best chromosome is picked and inserted into the offspring. Similar to crossover, mutation operation may produce an invalid chromosome, which is also fixed using the repair function. The mutation operation can help the optimization problem to jump out of the local optimization by sharing the global information about the population.

Cooling Schedule: One of the most important control parameter in the Selection Algorithm is •, called the Anneal Temperature; a parameter which is decremented, every time the system of particles approaches a better solution (or a low energy state). If $•_i$ be the initial temperature and $•_f$ be the final temperature, and t be the cooling time,

$$•(t) = •_f + (•_i - •_f)* •^t \qquad (6)$$

is our designed cooling schedule. This value of $•(t)$ is used in the Selection Algorithm. Here • is the rate of cooling, (usually, $0.7 • • < 1.0$) and t is the cooling time. For our purpose we considered t as the number of iterations. The Simulated Annealing algorithm incorporates the concept of probability through the Metropolis Acceptance Law into the optimal search ability of Genetic Algorithm.

Algorithm 6 *Cooling schedule*

***inputs*:** $\mathbf{t} \leftarrow$ the generation number
θ_f final temperature
θ_i initial temperature
$\alpha \leftarrow$ cooling coefficient

output: $\theta(t) \leftarrow$ Anneal temperature for the $\mathbf{t^{th}}$ generation

Procedure get_anneal_temperature(t) :

return $[\theta_f + (\theta_i - \theta_f)* \alpha^t]$

end Procedure

GROUP Algorithm

So far we have described different modules of the proposed algorithm. In this subsection, the proposed algorithm is accumulated as a whole and presented in a simple manner (in Algorithm 7)

Algorithm 7 *GROUP algorithm*

inputs: A set of N sensor nodes along with their position coordinates
output: An ordered sequence of the N nodes

Step 1: generate the initial population /* Algorithm 1 */

Step 2: for N times **do**:
Step 2.1: select parents /* Algorithm 2 */
Step 2.2: perform crossover /* Algorithm 3*/
Step 2.3: Evaluate offspring to select or reject it. /* Algorithm 4 */
Step 2.4: Repair selected offspring and perform mutation /* Algorithm 5 */
Step 2.5: Store the produced offspring for the next generation

Step 3: Mark the *best offspring* in the generation as **Cbest** *Increment* generation number by **1**
Step 4: Find the *anneal temperature* /* Algorithm 6 */
if $\bullet(t) \geq \bullet_t$ **goto** Step 2
else required sequence is represented by Cbest

Leader Selection Phase

Once the sub-optimal chain is formed we look for the node which has the maximum value of E_{resi}/D^4. Here E_{resi} denotes the residual energy of an individual node before starting a data gathering round and D is the distance of the base station from that node. The node with the maximum value of E_{resi}/D^4 becomes the leader. Here we consider the multipath fading (distance^4 power loss) channel mode, as the leader is concerned with communicating to the distant base station.

4. SIMULATION

Simulation Overview

To evaluate the performance of the GROUP scheme extensive simulations were performed on several random 100 node networks in a 50m*50m field as in [6]. Simulations performed in MATLAB show that GROUP scheme outperforms the data gathering schemes like PEGASIS [6] and The Scheme based on Ant colony Optimization (ACO) [7]. This readily implies the efficiency of our method over LEACH [4] and MTE [5]. As mentioned in Section III, for implementing our energy efficient data gathering protocol the chain formation was done by Genetic Algorithm with Simulated Annealing. Simulation results are shown in Table I. The base station was located at (25m, 150m) and energy per node was varied. As mentioned earlier, while comparing PEGASIS, ACO and GROUP schemes, a common threshold was introduced as the inter-nodal distance.

A second simulation was conducted in TOSSIM to study the *Packet Reception Ratios* for the three schemes. *Packet Reception Ratio* (PRR) is defined as the fraction of the number of packets received successfully out of the total number of packets required. The Simulation process in TOSSIM [11] considers the TOSSIM radio loss model, shown in fig. 2, which is based on the empirical data. The loss probability captures transmitter interference using original trace that yielded the model. More detailed measurements would be required to simulate the exact transmitter characteristics; however experiment has shown the model to be very accurate. In our experiment, we considered a fixed number of 20 packets for transmission from a network of 20 sensor nodes, where each node had a packet to send, to the Base Station. But due to the factor of packet loss, noise and unreliability of the wireless links, all the packets could not ultimately reach the Base Station successfully, if no retransmission attempts are made. Retransmission attempt simply means that when a node X sends a packet to another node Y, and the process fails, node X tries to resend the packet to Y. The maximum number of times this process can happen is called the *Maximum Retransmission Attempts* (MRAs). Our intension is to study the improvement of the PRR value at the Base Station with increasing MRAs for the different schemes. Thus this simulation helped us to take into account an interference model as if in a real life environment and a more realistic physical layer. Detailed results for the outcome are shown in Table II.

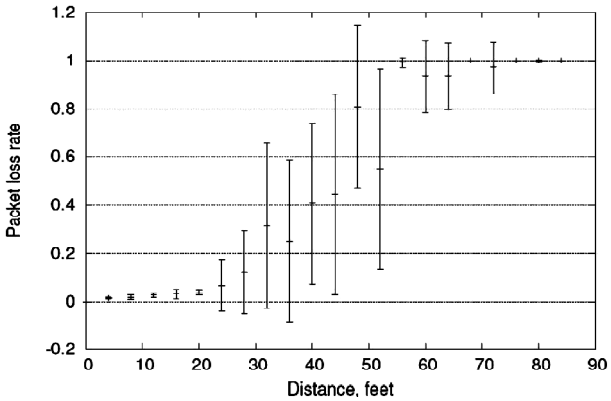

Figure 2: The Mean Packet Loss Rate Versus Distance is Shown, with Error Bars Indicating One Standard Deviation from the Mean. The Model is Highly Variable at Intermediate Distances. TOSSIM Radio Loss Model Based on Empirical Data

Simulation Results

In this section we show the results obtained in simulating our algorithm. Table I demonstrates the enhancement of network lifetime compared to the other schemes. From fig. 3, we find that GROUP largely outperforms PEGASIS, and also the chain obtained by ACO. It also reveals that GROUP largely performs both ACO and PEGASIS till about more than 50% of nodes in the network are dead. Networks with over 50% of nodes dead are very inefficient and therefore the degradation of performance of our schemes under these conditions can easily be ignored keeping in mind the superior performance with lesser percentage of dead nodes.

Table I
Number of Data Gathering Rounds for Various Schemes with Percentage of Dead Nodes

Intial Energy (J/node)		*Protocol*	*Percentage of Dead Nodes*					
			1	*10*	*20*	*30*	*40*	*50*
025	NUMBER OF ROUNDS	PEGASIS	811	887	1003	1024	1038	1050
		ACO	819	908	1037	1043	1064	1069
		GROUP	823	915	1045	1067	1085	1092
0.5		PEGASIS	1837	1971	2019	2044	2074	2082
		ACO	1850	1975	2057	2069	2109	2115
		GROUP	1861	1979	2089	2125	2163	2170
		PEGASIS	3724	3902	4049	4109	4174	4201
1		ACO	3731	3971	4133	4194	4223	4237
		GROUP	3735	4040	4203	4244	4257	4264

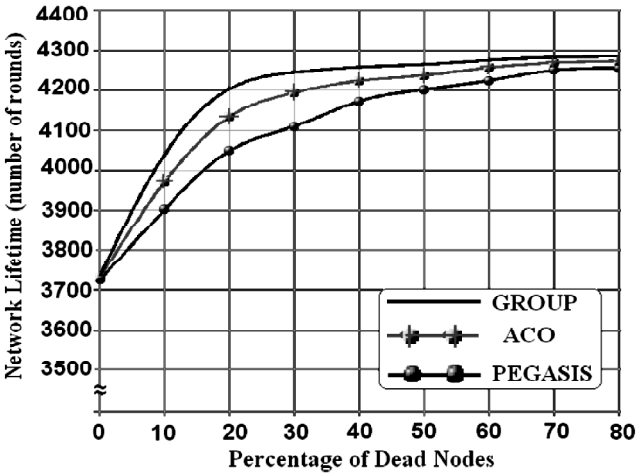

Figure 3: Performance Analysis of Different Protocols with Energy/ Node 1J and base Station at (25, 150)

Table II
Maximum and Minimum value of PRR with Increasing MRAS

Mra	*Group*		*Aco*		*Pegasis*	
	Min	*Max*	*Min*	*Max*	*Min*	*Max*
1	0.55	0.6	0.45	0.6	0.35	0.5
2	0.55	0.75	0.55	0.65	0.45	0.5
3	0.8	0.95	0.8	0.85	0.55	0.65
4	0.95	1	0.85	0.95	0.7	0.85
5	1	1	0.95	1	0.9	1
6	1	1	1	1	1	1

The packet loss rates have been compared in Table II. It shows how the PRR value at the Base Station increases faster in GROUP with increasing the retransmission attempts. In PEGASIS, certain links are comparatively much longer than the other links, which is responsible for the greatest number of packet losses. For a particular value of MRA the simulation has been conducted for 10 times. Here also, we see that packet loss in GROUP is less compared to PEGASIS or ACO schemes. Thus less number of retransmissions is necessary on the average for successful delivery of packets. This is also an important aspect of GROUP in terms of energy efficiency.

Figures 4.1 and 4.2 portray the chains formed by greedy algorithm and GROUP respectively, for the same node distribution in a 50m x 50m field. It depicts clearly how the inter-nodal distances are bound to increase when the greedy algorithm is used. The bold links in fig. 4.1 indicate the inter-nodal distances which are larger than the threshold. Thus as clear from fig. 4, the greedy chain introduces unreliable communication links in the network which will affect the average throughput of the network and in the process decrease the network lifetime too.

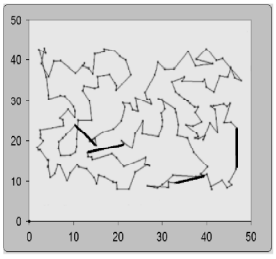

Figure 4.1: Greedy Chain

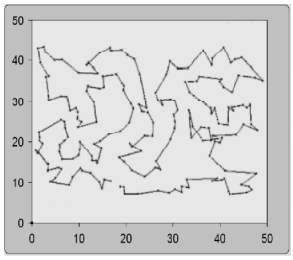

Figure 4.2: GROUP Chain

Hardware Feasibility

A lot of literature in the field of Wireless Sensor Networks has been devoted to the study of routing and data gathering protocols. Although much effort has

been given in the evaluation of their efficiencies and enumeration of their relative advantages, but most of the research publications in this context are more centered to theoretical proofs. While designing protocols it is an important criterion to justify the real-life feasibility of the scheme, particularly when the protocols are meant for the resource constrained sensor nodes. The sensor nodes contain limited amount of code and data memory, thus all schemes developed in the context of sensor networks should consider its feasibility for of an actual implementation. The GROUP was implemented in nesC [10], a component based dialect of the C programming language, meant to be hosted on the TinyOS software platform, an operating system which runs in majority of the sensor node modules including the Berkeley Mica Motes like Mica2 or MicaZ. The fruitfulness of the executable file produced by compiling a nesC program is that it can not only be used for the real hardware platforms but the same executable file can be tested and simulated in TOSSIM [11] environment. So, in absence of real motes, we simulated GROUP in the TOSSIM environment, which acts as a simulator for the TinyOS platform. Hence, the implementation and coding of the algorithm in nesC justifies the feasibility of our protocol for real hardware platforms, with limited memory and resources.

Since, this algorithm requires centralized knowledge about the sensor network, it would be best to carry out the algorithm in the Base Station and disseminate the result in the network before initiating data gathering tasks. However, this could well be dependent upon the application itself. In case, frequent communication with the Base Station is not feasible for all the nodes, this chain formation algorithm can also be applied in individual clusters in the sensor field, where these computations can be done by a local leader in each cluster. This will not only use up less resources as the number of nodes in a cluster is limited but also result in equalized energy dissipation among the local leaders. Secondly, this sort of distributed computation will speed up the process of self-organization of the network. Finally the Base Station could connect these local leaders to form the final optimized chain.

Recent advances in VLSI Technology, electronic design and fabrication and sophisticated tools have made it possible to design even hardware modules for simulating various heuristic techniques including Genetic Algorithms [13]. Hybrid hardware architectures incorporating Simulated Annealing in Genetic Algorithms has also been cultivated [14]. Much research has been done in this section, thus we hope, the algorithm for the formation of routing chain can be implemented in the hardware itself in near future.

5. CONCLUSION

Routing in sensor networks has attracted a lot of attention in the recent years and introduced unique challenges compared to traditional data routing in wired networks. In this paper we present a protocol that ensures that a near optimal energy utilization occurs thereby increasing network lifetime as is validated by simulation results. The application of Simulated Annealing helps to enhance the performance of our protocol. Reports of applications of using these meta-heuristic tools have been widely published, thus forming a solid background. Developing solutions with these tools offers two major advantages:

(i) Development time is much shorter rather than using more traditional approaches.

(ii) The systems are very robust, being relatively insensitive to noisy and/or missing data.

Moreover, GROUP has been coded in nesC, which justifies it to be feasible on real motes. Also, we have considered the TOSSIM interference model, while simulating packet loss rates for the various schemes. This simulation helped us to compare the reliability of the schemes for successful packet delivery, as if in a real life environment. All the results we obtained are in total compliance with our objective. Thus GROUP in true sense plays a good role in enhancing the lifetime of a sensor network by optimizing the routing paths.

References

[1] Clare, Pottie, and Agre : Self-Organizing Distributed Sensor Networks, In SPIE Conference on Unattended Ground Sensor Technologies and Applications, pp. 229–237, 1999.

[2] S. Lindsey, C. S. Raghavendra and K. Sivalingam : Data Gathering in SensorNetworks using energy*delay metric, In Proceedings of the 15th International Parallel and Distributed Processing Symposium, pp. 188-200, 2001.

[3] Yunxia Chen and Qing Zhao : On the Lifetime of Wireless Sensor Networks, Communications Letters, IEEE, Volume 9, Issue 11, pp. 976–978, DigitalObject Identifier10.1109/ LCOMM.2005.11010., 2005.

[4] W. Heinzelman, A. Chandrakasan, H. Balakrishnan : Energy- Efficient Communication Protocol for Wireless

Microsensor Networks, *IEEE Proc. of the Hawaii International Conf. on System Sciences*, pp. 1-10, 2000.

[5] T. Shepard: A Channel Access Scheme for Large Dense Packet Radio Networks in Proceedings of ACM SIGCOMM Conference, pp. 219-230, 1996.

[6] S. Lindsey, C. S. Raghavendra: PEGASIS: Power Efficient Gathering in Sensor Information Systems", *In Proceedings of IEEE ICC* 2001, 1125-1130, 2001.

[7] Ayan Acharya, Anand Seetharam, Abhishek Bhattacharyya, Mrinal Kanti Naskar, Balancing Energy Dissipation in Data Gathering Wireless Sensor Networks Using Ant Colony optimization, 10th International Conference on Distributed Computing and Networking-ICDCN 2009, pp. 437-443, 2009.

[8] D. Goldberg, B. Karp, Y. Ke, S. Nath, and S. Seshan. : Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, 1989.

[9] Kirkpatrick S. : Simulated Annealing, *Sci*., **220**, 1983.

[10] D. Gay, P. Levis, R. von Behren, M. Welsh, E. Brewer, and D. Culler. The nesC language: A holistic approach to networked embedded systems. In Proc. Programming Language Design and Implementation (PLDI), 2003.

[11] P. Levis, N. Lee, M. Welsh, and D. Culler. : TOSSIM : Accurate and Scalable Simulation of Entire TinyOS, In Proc. the First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003), November 2003.

[12] Zhi-Feng Hao, Zhi-Gang Wang; Han Huang. : A Particle Swarm Optimization Algorithm with Crossover Operator, *International Conference on Machine Learning and Cybernetics* 2007, 19-22.

[13] Tachibana, T., Murata, Y., Shibata, N., Yasumoto, K., Ito, M. : A Hardware Implementation Method of Multi-Objective Genetic Algorithms, *IEEE Congress on Evolutionary Computation*, 2006. CEC 2006., pp. 3153-3160, Sep. 2006.

[14] Masaya Yoshikawa, Hironori Yamauchi, and Hidekazu Terai, : Hybrid Architecture of Genetic Algorithm and Simulated Annealingr, Engineering Letters, *International Association of Engineers*, **16**(3), 2008.