Energy and Environmentally Aware Multi-path Routing Protocol Using a Genetic Algorithm in a Wireless Sensor Network

Abhinay Devapatla
Department of Computer Science
Michigan State University
East Lansing, MI, USA
devapatl@msu.edu

Alec Said
Department of Computer Science
Michigan State University
East Lansing, MI, USA
saidalec@msu.edu

Nicole Schneider
Department of Computer Science
Michigan State University
East Lansing, MI, USA
schne542@msu.edu

***Abstract*— Wireless Sensor Networks (WSNs) play an important role in several applications, such as environmental monitoring, industry automation and health care. A new routing protocol, called Energy and Environmentally Aware Multi-path Routing Protocol Using a Genetic Algorithm (EEAMRP-GA) for which is proposed in this paper, aims at not only optimal energy consumption in the WSNs, but considers the environmental factors that effect a network as well.**

EEAMRP-GA uses a Genetic Algorithm (GA) to search in a smart way for energy saving and environmentally safe routing paths. For this purpose, the routing protocol generates a set of possible routes and then using a combination of selection, crossover and mutation processes, the GA generates a list of possible efficient routes. While generating these possibilities, it considers three factors: the distance from the source to sink node in a route, the energy consumed by a route, and the possible danger a route may encounter.

This protocol was put forward to improve the networks' resilience and adaptability. It does this by distributing energy more evenly across sensor nodes and chooses paths with a better chance of safely delivering the data. From the research so far, the newly proposed routing protocol works best in smaller-scale networks compared to the existing ones in terms of energy consumption and how long it can stay alive.

Overall, EEAMRP-GA presents a promising approach towards achieving energy-efficient and environmentally friendly routing in WSNs, contributing to the development of sustainable networks for various applications.

Keywords—Multi-path routing protocols, Energy Efficiency, Genetic Algorithms.

I. Introduction

Creating an ad hoc network with adequate power efficiency and environmental consciousness marks a critical field in research and development. An ad hoc network is a way to connect nodes without an infrastructure device. There are different applications of ad hoc networks including mobile ad hoc networks (MANET), which are used for human interactions and wireless sensor networks (WSN), which are used for environmental factors. WSNs are finding their way into many applications including environmental sensing, industrial automation, threat detection and much more; thus, the optimization of power consumption is crucial for reducing its environmental footprint.

There are a few components that go into creating a WSN with the above aspects. In terms of this research, that includes a routing protocol and a genetic algorithm.

A routing protocol is created to specify the rules needed to designate how a router distributes data packets among a network. Routing protocols can be either static, using preconfigured routes to forward data to its destination, or dynamic, using algorithms to find the best path from source to destination. There are also two main types of routing protocols. Interior Gateway Protocols (IGP) operate within an Autonomous System (AS). Exterior Gateway Protocols (EGP) operate between an AS. Specifically looking into IGPs, there are three types: Distance Vector Routing Protocols, Link-State Routing Protocols, and a Hybrid of the previous two options. Distance vector routing protocols send complete updates, entire routing tables, to neighbors periodically while link-state routing protocols only send updates of the topology when there is a change in the network, but to every router in the network.

Multi-path routing protocols can be created to find multiple routes from source to destination in the network so that if a route becomes unavailable, there are other options available. This is done during a single route discovery process. Once the routes are collected, they can then be used in a genetic algorithm to find the optimal route for packet delivery.

Genetic algorithms (GA) are a type of evolutionary computation method. They solve optimization and search problems by using the process of natural selection. GAs are made up of the following steps: initialization, fitness calculation, selection, crossover, and mutation.

Initialization of the population occurs by creating individuals, also called chromosomes. Chromosomes are filled with genes, which are the representation of the problems parameters. Then, each individual from the population receives a fitness score obtained by sending it to the fitness function. The next step is to perform selection, where the parents are selected for recombination. Selection can be done using different strategies such as elitism, where some number of individuals obtaining the best fitness score are selected, or tournament, where the best fitness scored individual is selected from a tournament of size k. Following this, crossover can be performed. The parents from the previous section are paired and their genes are combined to obtain children. This can be done using one point, multi point, or uniform crossover. The resulting children move onto the next step of possibly being mutated. This is usually done at a small rate to help maintain previous favorable genes that already exist. Finally, the termination criteria can be checked. This could include reaching the maximum generation number or reaching the minimum criteria needed. If not terminated, the algorithm starts its next generation with its new population of individuals; however, if termination criteria are reached, then the algorithm ends, and the optimal solution and results can be reviewed.

One factor that commonly used algorithms, both genetic and not, do not account for is the environmental factors that can affect the nodes within a system. In this paper, a new approach is proposed – an Energy and Environmentally Aware Multi-path Routing Protocol with Genetic Algorithm (EEAMRP-GA). The goal of EEAMRP-GA is to apply genetic principles towards resolving energy and environmental issues that WSNs face, making it promising for sustainability and efficient data transmission.

The project report is laid out as follows: Related Work explains existing work and its limitation in this area, Methodology discusses how the research was conducted, Design shows what was used to perform the methodology, Technical Results and Analysis looks at the research findings, Future Work considers what further research and work could be done, Summary and Lessons Learned summarizes what was taken from this research, and References lists the papers and journals utilized.

II. Related Work

The routing algorithm proposed in *Energy Efficient multipath routing algorithm for Wireless Multimedia Sensor Network* [1] used an efficient multipath routing based on GA (EMRGA) to "select the best path based on the cost function with the minimum distance and the least energy dissipation" [1]. The routing algorithm combined cluster formation, cluster head selection, and multi-path routing to reduce energy consumption during data transmission. The Wireless Multimedia Sensor Network (WMSN) nodes were represented using a weighted undirected graph. The sensor nodes were randomly deployed into a field. The authors developed an energy function based on the number of hops. The routing algorithm formed clusters of sensor nodes within a certain radius threshold and assigned a node (based on the distance to the center and energy level) as the cluster head. The multipath genetic algorithm fitness function determined the most energy efficient path by calculating the distance to the sink and cluster heads and the energy function. The algorithm was compared against existing routing protocols but was limited to energy consumption performance. Overall, EMRGA had better energy consumption, but no studies were done to compare additional metrics, such as packet loss.

Another important research paper referenced was *Environmental-fusion Multipath Routing Protocol for Wireless Sensor Networks* [2]. Overall, this paper's goal was to extend the network lifetime of wireless networks that have been placed in dense wilderness or remote locations. Nodes that have been deployed to locations like this can suffer from the adverse conditions of nature, therefore this study proposed to specifically route around sections of the structure that were under environmental threat. These results acted as a valuable overview of what environmental factors are important to consider and how they might be included in a routing protocol. This paper focused on considering the measured environmental readings of the nodes themselves and used this information as a parameter within their routing protocol along with residual energy and node depth. Each of these environmental factors were defined by graphs representing the optimal operation range, and these fields were combined to represent a multi-filed range. These track if any nodes within the system occupy "danger zones" that are not safe for packet deliveries to route through. This, with additional traffic and routing allocation mechanisms, resulted in significant improvements to the packet success rate and the system's lifespan. The limitations of this paper are few as it was very comprehensive overall, but it does not consider the useful application of its environmental considerations to that of smaller networks making use of genetically defined routing protocols.

One more previous work that was looked at was, *Multipath Routing Protocol Using Genetic Algorithm in Mobile Ad Hoc Networks* [3]. This work looked to find the optimal solution to two issues: energy consumption and packet loss. They propose adding a new fitness function (FFn) and GA to use in conjunction with the existing Ad Hoc On-demand Multipath Distance Vector (AOMDV) routing protocol. The AOMDV will find the initial routes through the network and the FFn and GA will provide the optimal solution found after completing the algorithm. The research not only shows the results of the AOMDV-GA but also the AOMDV-FFn, where it stops after finding the fittest routes and doesn't continue to crossover and mutation. Using these two main algorithms, AOMDV-GA and AOMDV-FFn, they were able to outperform other routing mechanisms in terms of network performance, the former being slightly more efficient than the latter.

III. Methodology

The goal of this project is to design a routing protocol that utilizes a genetic algorithm to find efficient routes while also including environmental consideration that many routing protocols ignore. Using a simulation of a node network system used to gather data this project hoped to design a new routing protocol using a genetic algorithm (EEAMRP-GA) that considers the safety level of the nodes within a route, the residual energy level, path congestion, and the distance traveled when routing to the designated sink node. The goal is to measure the performance of EEAMRP-GA compared to other routing algorithms.

EEAMRP-GA is based in a simulated system and therefore the measurement of true packet loss or packets dropped is not the best way to measure the success of this routing algorithm. This is because while a dead or incomplete route can mimic a dropped packet, this is at a much smaller scale than a real-life system. Therefore, similar to AOMDV-GA, the best way to measure the life span of this network is to monitor the number of dead nodes and total residual energy of the nodes as signals. are sent through the system. When looking at the results, it is better to have the least number of dead nodes in the system after data transmission and when compared to AOMDV-GA, A Star, and Distance Vector Routing, one can see which of these methods of routing puts the least amount of strain on the node network. The goal here is that while the other routing algorithms might be faster or spend less energy choosing an ideal path than EEAMRP-GA, it will still be able to prolong the life of the system by including factors such as danger level indicated by the environment. While some related works include factors such as environmental danger, like Environmental-fusion, these do not consider the benefits of using a genetic algorithm to function as a multi path consideration rather than an older ad hoc algorithm. By combining the multipath considerations and a new fitness evaluation for each of those paths, EEAMRP-GA will be a slower but more efficient routing method and will prolong the life span of the node network system, which is most important while monitoring an area using this technique.

The development of the network simulation involved setting up strict parameters while also incorporating a level of randomness to the system to mimic real life. The simulation had defined system dimensions and a fixed number of starting nodes defined at initialization, but the placement of nodes, danger zone regions, and natural energy dissipation and initial node deaths were randomly generated. Additionally, throughout the system the danger zones would randomly shift to mimic dynamic changes to the environment that range from as serious as a forest fire spreading to as simple as shifting temperature levels. The goal behind designing the simulation network described was to create a blank slate over which any routing protocol could be tested. The nodes and danger zones are pseudo random with the use of a seed so that tests can be more accurately compared. This reproducible randomness allows multiple algorithms to be tested on identically random networks to minimize fluke results or large fluctuations in network lifespans.

When developing the GA portion of the routing protocol, there were many aspects to consider including what type of crossover, mutation, and selection to use as well as what to consider in the fitness function. Regarding the operators used in the GA, tournament selection is used first to find the two parents to create a child from. Keeping a relatively low tournament size to increase the selection pressure, or the degree to which better individuals are favored. Following this selection, one-point crossover is used to create a child and was selected due to wanting to make sure that the paths are not too diverse, therefore ensuring viable paths were being created. Next, each child has a possibility of being mutated. Initially, the mutation created would randomly select a new node to be added to the route at a random spot. This meant that when the new node was inserted, it had to recalculate the route to make sure it was still viable. Most of the time it had to add extra nodes in between the old nodes that were next to the new node to connect them. Due to the complexity of this, another mutation operator was created to test against it. In the new mutation, a random node was selected for deletion. Then the nodes right before and after the deleted node were connected. While this also resulted in sometimes adding additional nodes to connect them, it created less complicated and therefore better routes. The second mutation operator is the mutation being used in the EEAMRP-GA. Lastly the GA uses elitism selection to create the new population. This is used to make sure the population size does not get out of control.

Regarding the fitness function created in the GA, it considers 4 properties. It calculates the ideal distance from the source to the destination of the network, D_i, and divides that by the actual distance of the route, D, in question. It calculates the total energy of the route, E, and divides that by the ideal energy of the route, E_i, which would be if all the nodes had full batteries. It calculates the congestion of the route, C, and divides that by the ideal congestion, C_i. For this calculation, the ideal congestion needed to be 1 since if it were 0, each congestion calculation would result in 0. The actual congestion is calculated by taking the actual distance of the route and dividing it by the propagation speed of the medium, 3X10^8. This calculates the propagation delay, which is a part of the end-to-end delay, which contributes to congestion in a network. This was the only viable way to add congestion into the network with the simulation that was created. Lastly, it calculates the safety of a route, S, by adding up the total number of nodes in danger zones and divides it by the ideal safety, S_i, which is the total number of nodes in the route. By combining these four calculations, the fitness score for each route can be calculated.

$$fitness = \left(\frac{\left(\frac{D_i}{D}\right) + \left(\frac{E}{E_i}\right) + \left(\frac{C_i}{C}\right) + \left(\frac{S}{S_i}\right))}{4} \right) * 100$$

Equation 1: Fitness function used in GA to produce ratio of fitness scaled from 0 to 1.

Parameters	Values
Number of Nodes	200
Initial Energy	100 J
Range	10
Number of Generations	50
Population Size	100
Offspring Size	20
Crossover Rate	80%
Mutation Rate	10%

Table 1: Parameters of EEAMRP-GA

IV. Design

Network Simulation

The first step of testing the proposed genetic algorithm was to design a simulated node network system to test any routing protocol. The goal of this network simulation was to mimic a network system that might be used to monitor a forest or section of dense wilderness that cannot easily be accessed for other modes of data collection. In cases like this, scattering a collection of affordable and simply designed nodes through the area to collect data to be sent to a sink node is common. These nodes are often scattered from planes, as flying above them is the best way to get coverage through rugged terrain. So first, a node class was created to act as the simple data-gathering devices generally used in these types of networks. The Node class was constructed to hold location data in the form of a set of (x, y) coordinates, a battery level that starts at 100% and represents a capacity of 100 joules**,** a range representing how far the node can communicate, a list of neighboring nodes withing sed range which is initialized initially as an empty list, and an id. These values are used to mimic the operation of an actual node, which would have basic battery monitoring and signal range capabilities that would be used to form and maintain the node network system. Because this node class is used in a simulation, extra information is stored to monitor the general state of each node within the system. These were three Booleans, which represent special information about the node: one named dead, another named danger, and a third named sink. All three of these Boolean values are initialized as False, as the default node should begin as a living node, not in danger, or set as the sink node. This way, as nodes are found to be dead or in danger, this value can be adjusted accordingly. Now that the node class has been set up to represent the monitoring devices used within a forest, the space they occupy must be defined.

To mimic a space for these nodes to occupy, a class named GridSpace was created. This class allows for the creation of a GridSpace object that represents a two-dimensional coordinate space for the nodes to be represented as points. This object is initiated with an initial width and height, which are the bounds of the graph space. Within this project, a width and height of 100 was used for testing. This object also holds a list named nodes, which is the list of "devices" that occupy a position within its coordinate graph, a list named danger_zones, the number of nodes in the system, and the name of the current sink node. This graph space acts as the forest on which the node network system exists. Using a function named populate_grid, the number of nodes, node range, and node success ratio are parameters and used to scatter nodes across the GridSpace object. The input number of nodes requested is created with unique random (x,y) coordinates and the specified range. These nodes are recorded within the self.nodes list of the Gridspace; with this, the "forest" space being monitored now has nodes scattered randomly across its space. To mimic the dangerous nature of dropping these nodes, the success ratio kills 10% of the initial nodes dropped by marking node.dead as true. This represents the possibility of nodes dropped from a plane becoming damaged during placement or landing in an unideal location, such as in water or vertically out of range of other nodes. With the GridSpace populated, the nodes now need to be connected using the function connect_network; this function uses nested for loops to step through each node and find its distance to every other node in the system: if this distance is less than or equal to the node's range then the nodes are considered to be adjacent to each other. The second node is added to the neighbors list of the initial node so that they can communicate directly. The final step is to pick a random node within the system to operate as the destination for data. A single sink node is selected from the system so that the simulation can test communication from any node to any other node in the system, rather than a cluster system with multiple sinks because this could lead to heavy traffic in small sections of the graph and skew testing around dead zones caused by this traffic. The graph now has a fully connected node network system. The final stage in building this network simulation is to define danger zones using the add_ddanger_zones function from the DangerZones class, which inputs a number of zones to be added and an optional maximum size for these zones. This class randomly defines ellipses within the given parameters to be added to the GridSpace list called danger_zones. This way, each network will have a list of ellipses and their graph coordinates. If no maximum size is given, then the zones will use the size of the graph itself as a guide. The rest of the functions within GridSpace are used to maintain or plot the network. The new_sink function defines the nearest living node as the new destination for data collection if the current sink node dies. The check_danger function steps through each node within a route that sends a signal, and if any nodes within the route are in danger, then there is a 5% chance that the node will malfunction due to dangerous conditions and die. This simulates the possibility of high humidity, temperature, or other natural phenomena affecting the node and causing it to be damaged. The function node_status is the primary maintenance function used to check each node in the system for new information. If the danger zones of the system have changed, then it adjusts the node.danger Boolean to true according to the new coordinates that are now in danger, and the value of nodes no longer in danger to false. This function also checks the battery level of each node and marks any node with a battery level less than or equal to zero as dead. For a now operational simulated node network system, there is a function named send_data that mimics the transfer of a signal through a selected route. This function updates the battery level of nodes by decreasing it between 1 to 2 Joules, as this is the average energy used to send a single from nodes similar to the ones we are mimicking in our system. The final function, gen_routes, exists to support the genetic algorithm designed to route through this system and is described below.

The result of these network systems can be seen **Figure 1**, this is before any signals are sent along paths within the system. The blue nodes represent living nodes that are active within the system. As discussed, they are randomly distributed across the grid space to mimic nodes through a forest. The red ellipses are the danger zones representing weather or conditions threatening the system. The red nodes within these zones are marked as being in danger and have a chance of being killed by the zone. Then, **Figure 2** represents the progress of a system that has sent 100 signals from source to sink. This system shows a nearly dead system but represents the functionality of this simulation. As nodes die the connections are redefined and as time passes the danger zones shift and update as well.

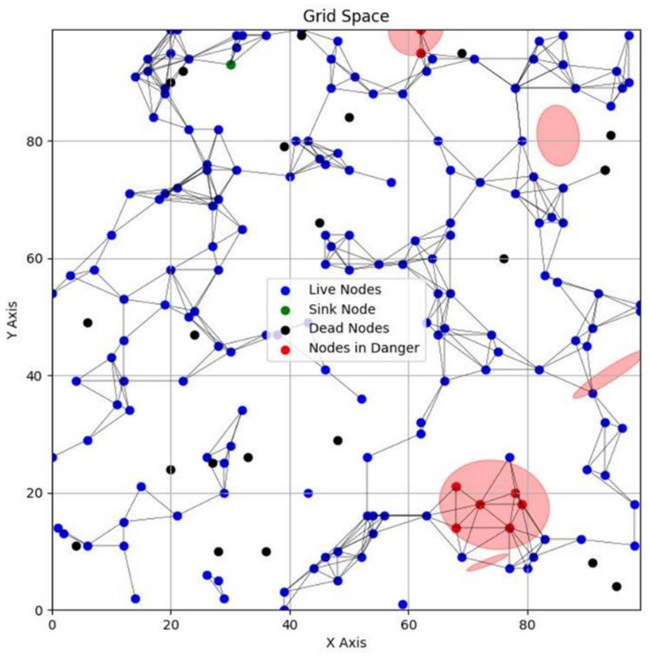

Figure 1: Fresh node network system ready for a routing protocal to be tested

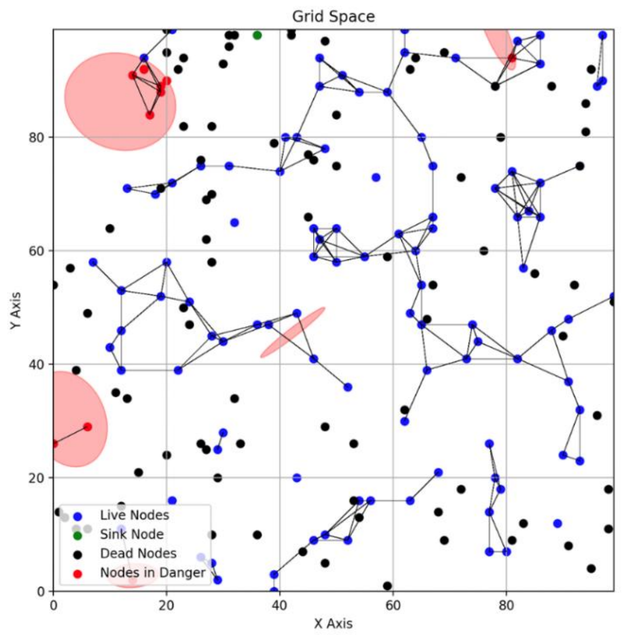

Figure 2: Nearly dead node network system after sending data.

Genetic Algorithm

The representation of this genetic algorithm design depends on the routes generated operating as the member of the population, and each of the nodes within these routes represents the individual genes themselves. For example, a route that contains nodes 1, 2, 3, 4, and 5 would be a single member, and each node within this member's chromosomes is a singular gene. This becomes clear in the function gen_routes, which form a population to feed into the genetic algorithm. This function inputs a population size and a source node. Using a depth first search function, named unique_dfs, which has been adjusted to only produce a set number of unique paths that route from the source to sink nodes equal to the requested number, the population size is generated. These routes make up the population of the genetic algorithm will use. In order to find the best possible path from a random source to the dedicated sink, a population is created and input into the genetic algorithm function along with the maximum number of generations, offspring per generation, the network node list, mutation rate, and crossover rate. For each generation, a number of "children" route equal to the input offspring are produced. This reproduction starts with tournament selection, where 5 random routes are selected from the population. Of those members of the population, the best is selected to be one of two parents for two new children. The best member is chosen by sorting each route according to its fitness value. An equation within the get_fit function defines each member's fitness, which observes each node's battery level, location, and danger values to provide a score scaled from 0 to 1. The first part of this fitness equation is the residual energy remaining within the route. Each gene's residual energy is summed up and divided by the maximum possible energy to produce the average remaining energy for the member. The closer to 1 this battery level is, the better this section of the fitness score will be. Similarly, each node currently marked as not being in a danger zone is tallied up and divided by the total number of nodes within the route. This represents how safe this member of the population is, same as energy, the closer to 1 this number is, the better. Next, the distance the single needs to travel along this route. The shortest possible distance between source and sink is found using the quadratic formula; this value represents a direct path from origin to destination, which is divided by the sum of distances from each node to the next within the path this member represents. The shorter the sum of distances is the closer it will be to the ideal distance, and the closer it will be to 1. Once the two parents have been selected for reproduction, a single point crossover takes place by randomly selecting a gene within parent 1 and searching for that gene within parent 2; if the parents share this gene, then it is selected as the crossover point, and the children are produced. This method is implemented to preserve the viability of each member's path; by using common genes as a crossover point, both children will be usable routes. This process can be seen represented in **Figure 3**. If this randomly selected gene is not in both parents, then a new gene will be selected until a common connection is found. If no genes are shared between the parents, then new parents are selected with two new tournaments.

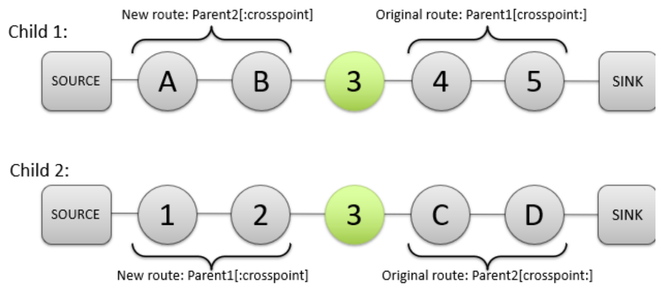
Child 1:
New route: Parent2[:crosspoint]
Original route: Parent1[crosspoint:]
SOURCE
A
B
3
4
5
SINK
Child 2:
SOURCE
1
2
3
C
D
SINK
New route: Parent1[:crosspoint]
Original route: Parent2[crosspoint:]

Figure 3: Crossover preformed on parent1: 1 2 3 4 5 and parent2 A B 3 C D.

After the crossover, a single point mutation takes place. For each of the new children a random gene is selected to be deleted, the gene directly before and after this deletion are then entered into a depth first search and are reconnected. This process is represented within **Figure 4** which shows a gene being selected and replaced. The benefit of this mutation is the reasonable level of diversity it introduces to each chromosome. This new route is not overly mutated as to not introduce too much diversity into the system and disrupt the evolutionary process rather than assisting it.

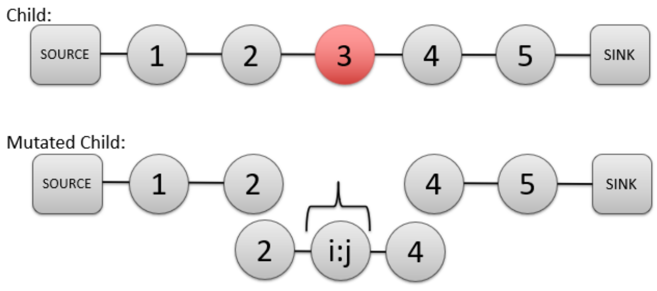

Figure 4: Deleton mutation.

Within this system there is an 80% chance of crossover and a 10% chance of mutation. After the children are produced, the offspring are all added to the population, which is sorted by fitness, and the worse chromosomes outside of the population size are discarded. This repeats for as many generations that were requested before a sorted list of routes is returned and used as the multipath routing options.

Network Lifespan

To test this new routing genetic algorithm 100 “data signals” were sent from 100 randomly selected nodes to the dedicated sink node and the resulting network was observed. The premise behind this testing is to run the same or similar random danger zones for each data packet simulation sent through the system and after testing different levels of signals sent through variously sized and populated systems, the performance of the EEAMRP-GA can be evaluated, along with other algorithms. These metric of success relite on the prediction that other routing algorithms not taking into consideration that the danger zones will disrupt the life span of the system.

V. Technical Results and Analysis

The metric of success used to compare the EEAMRP-GA algorithm’s network lifespan was the number of dead nodes over many iterations and remaining energy value over many iterations. The EEAMRP-GA algorithm was compared against the greedy algorithm A* search and the Distance Vector Routing Protocol for performance comparison. The simulation was run at a various number of nodes to collect a comprehensive overview of the EEAMRP-GA algorithm. **Figures 5 – 8**, below show the graphs associated with a network of 250 nodes. Specifically at 250 nodes, the EEAMRP-GA performs 14.809% better than A* search and 14.809% better than Distance Vector Routing protocol at energy consumption over 100 iterations. At 200 nodes, the EEAMRP-GA performs 5.232% better than A* search and 5.222% better than Distance Vector Routing protocol at energy consumption over 100 iterations. At 300 nodes, the EEAMRP-GA performs 12.484% better than A* search and 12.488% better than Distance Vector Routing protocol at energy consumption over 100 iterations. Additionally, with better remaining energy values there were a smaller number of dead nodes on average compared to A* search and Distance Vector Routing protocol. At 200 nodes EEAMRP-GA on average had 0.85 less dead nodes than A* search and 0.85 less dead nodes than Distance Vector Routing Protocol. At 250 nodes EEAMRP-GA on average had 4.86 less dead nodes than A* search and 4.58 less dead nodes than Distance Vector Routing Protocol. At 300 nodes EEAMRP-GA on average had 7.19 less dead nodes than to A* search and 7.2 less dead nodes than Distance Vector Routing Protocol. The results indicate that for small scale node network systems the EEAMRP-GA is better at maintaining network life span. However, it is only more efficient until scalability of the network becomes an issue i.e. large node networks and multi-sink problems.

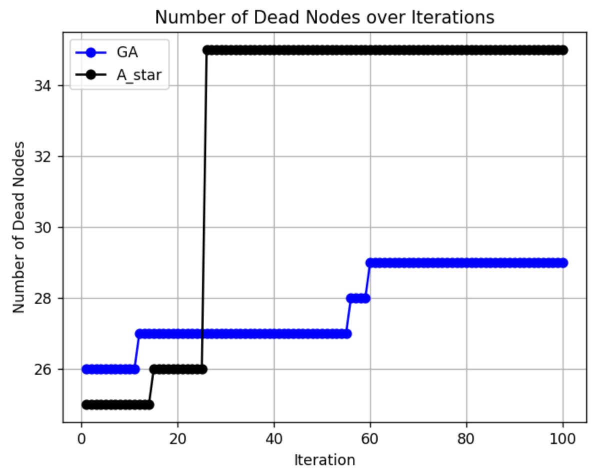

Figure 5: Number of dead nodes in EEAMRP-GA and A_star search

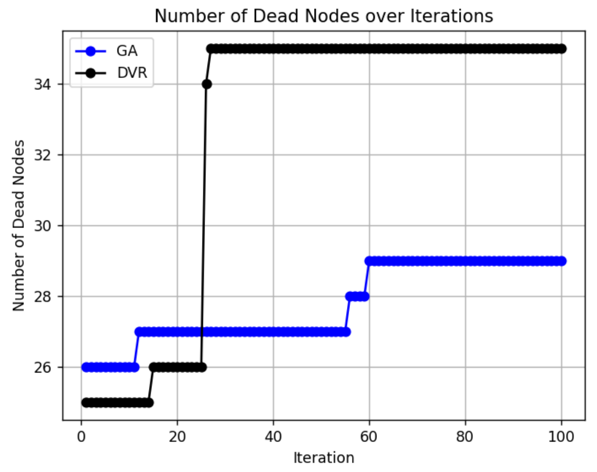

Figure 6: Number of dead nodes in EEAMRP-GA and Distance Vector Routing Protocol

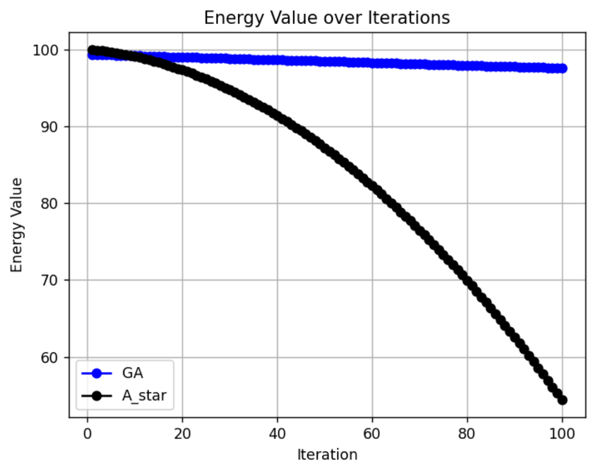

Figure 7: Amount of energy remaining in EEAMRP-GA and A_star search

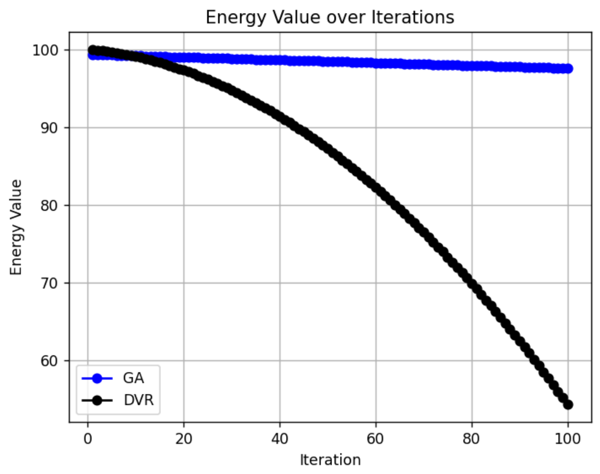

Figure 8: Amount of energy remaining in EEAMRP-GA and Distance Vector Routing Protocol

VI. Future Work

In the future, the use of a multi-sink node cluster system should be investigated to study its effects on scalability. This is because the larger a system was through the testing of this project the worse the performance was. Shockingly with only a node network of 500+ nodes the performance dropped by nearly 60% resulting in a worse outcome compared to the A Star method. This scalability issue could also be solved by adjusting the GA itself and assigning weighted values which make some aspects of the fitness function more important than others. This was a stretch goal for the project but due to the time constraints of the semester it was left inadequately explored. This would not improve the speed scalability aspects of this projects limitations, but it could improve the network life span which is also valuable. Additionally, other search algorithms other than a custom Depth First Search should be utilized in the Genetic Algorithm to generate a population. Lastly, the computational complexity of this system is high, so future work must prioritize researching ways to optimize the genetic algorithm.

VII. Summary and Lessons Learned

In summary, this project indicates that there is a lot to be improved upon when it comes to improving genetic algorithms for the use of multi path routing protocols. While it is an efficient method of finding many viable paths all ranked in order of efficiency it is difficult to obtain results in a timely matter. Especially as the node network system itself grows. To accurately test the results of this project the number of generations and route population needed to remain relatively low so as to not overwhelm the find routes function and take an unrealistically long time to find the best route and send the signal. If the point of this algorithm was to protect node network systems from the natural environment, they inhabit then there is no realistic way that this Genetic Algorithm could respond quickly enough to new data needing to be accounted for. While the scalability issues described here mean that this project was limited to small scale success this does not mean that future works cannot continue to improve upon these findings. As mentioned in future works there are options to improve upon this type of Genetic Algorithm. Making use of multiple sink nodes to maintain control over clusters of "servant" nodes could help to improve the scalability of this system. An alternative search function for route production could help to improve the speed at which the algorithm can complete the requested generations. The current modified depth first search while efficient within small networks with an even distribution of nodes has trouble finding unique paths as the network begins to fail. As a final through this project successfully achieved a life span about 7% better than A Star, but this unfortunately does not surpass the average 15% achieved by *Energy Efficient Multipath Routing Algorithms for Wireless Multimedia Sensor networks.* At the end, the EEAMRP-GA provides many benefits but at the same time as room for improvement.

VIII. Workload Distribution

The workload for the project report was evenly distributed among the group members. While the sections were split evenly amongst the group, everyone worked on each section be proof reading and adding material where needed. Alec worked on developing the node network simulation system that the algorithms are tested within and assisted in the creation of the Genetic Algorithm by attempting to test weighted values and defining the new single point crossover and modified depth search. Nicole formatted and designed the Genetic Algorithm and tested different formats for the mutation and tournament as well as various input values. Abhinay helped define the node network representations as well as formatted the mathematical representation of results and plots.

IX. References

[1] Genta, K.Lobiyal, and Abawajy, "Energy efficient multipath routing algorithm for Wireless Multimedia Sensor Network," *Sensors*, vol. 19, no. 17, p. 3642, Aug. 2019. doi:10.3390/s19173642 https://www.mdpi.com/1424-8220/19/17/3642

[2] X. Fu, G. Fortino, P. Pace, G. Aloi, and W. Li, "Environment-fusion multipath routing protocol for wireless sensor networks," Information Fusion, vol. 53, pp. 4–19, Jan. 2020, doi: https://doi.org/10.1016/j.inffus.2019.06.001

[3] A. Bhardwaj and H. El-Ocla, "Multipath Routing Protocol Using Genetic Algorithm in Mobile Ad Hoc Networks," in IEEE Access, vol. 8, pp. 177534-177548, 2020, doi: 10.1109/ACCESS.2020.3027043. https://ieeexplore.ieee.org/abstract/document/9206583

[4] Y. Song, C. Gui, X. Lu, H. Chen, and B. Sun, "A genetic algorithm for energy-efficient based multipath routing in wireless sensor networks," *Wireless Personal Communications*, vol. 85, no. 4, pp. 2055–2066, Jul. 2015. doi:10.1007/s11277-015-2891-3. https://link.springer.com/article/10.1007/s11277-015-2891-3

[5] S. Kumar Singh and C. Asbe, "Evaluation of Network Lifetime by Multipath Routing Protocol for WSN," *2023 IEEE 4th Annual Flagship India Council International Subsections Conference (INDISCON)*, Mysore, India, 2023, pp. 01-07, doi: 10.1109/INDISCON58499.2023.10270261. https://ieeexplore.ieee.org/abstract/document/10270261

[6] S. Kim, H. Roh, and K. Jung, "Cluster-disjoint multipath routing protocol for real-time and reliable packet transmission in wireless sensor networks," *Sensors*, vol. 23, no. 21, p. 8876, Oct. 2023. doi:10.3390/s23218876. https://www.mdpi.com/1424-8220/23/21/8876