Building Paths to Reduce Latency and Increase Resilience to Cyberattacks

Kevin Olenic
Department of Computer Science
Brock University
St. Catharines, ON, Canada
ko19af@brocku.ca

Michael Dubé
Department of Mathematics
University of Guelph
Guelph, ON, Canada
mdube04@guelphu.ca

Sheridan Houghten
Department of Computer Science
Brock University
St. Catharines, ON, Canada
shoughten@brocku.ca

***Abstract*—Due to the constant need for internet access, the multitude of requests from various user devices for content from multiple sources causes cell towers to suffer high latency if data transmissions are mismanaged. Moreso, if an attack such as DoS or DDoS is launched on a network with mismanaged data transmissions, the result could be catastrophic, thus making proper management of transmissions paramount. Various techniques for managing these communications exist, but they are computationally costly and only account for select variables such as energy consumption or the amount of data transmitted. We use a genetic algorithm approach to generate network configurations with reduced latency based on multiple factors: transmission distance, data transmitted over a connection or to a node, and energy consumption, with the eventual goal of these networks having increased resilience to cyberattacks.**

***Index Terms*—genetic algorithm, 5G networks, cybersecurity, network design, latency**

I. Introduction

Wireless network usage has skyrocketed in the past few decades as more individuals use them for personal entertainment and in their professional occupations to get information. This has caused widespread deployment of wireless cell towers to receive and transmit data from various sources worldwide, opening up numerous issues. Two such issues that plague these network communications are (i)m the sheer amount of requests and data the cell towers must manage, and (ii) the threat of malicious individuals launching Denial of Service (DoS) or Distributed Denial of Service (DDoS) attacks on these networks. Due to the complex nature of such networks, it is both computationally expensive and time-consuming to perform a complete simulation of the network, testing different configurations to determine the optimal connection layout for a set of towers. Further complicating the matter is the fact that multiple factors must be considered when designing a network communication layout. These additional aspects increase the complexity of simulations, causing most efforts to focus on a limited number of factors affecting latency.

In this work, we present a solution in the form of a genetic algorithm to design the connection layouts of networks to minimize latency and build their resilience against DoS and DDoS attacks, considering multiple fitness functions independently. Further investigations can then perform simulations on the networks to determine their resilience to DoS/DDoS attacks causing some connections or nodes to be unavailable or lost.

The remainder of this paper is organized as follows. Section II discusses related work, while the methodology is presented in Section III. The results are presented in Section IV. Section V concludes the paper and discusses possible future work.

The data and code used in this work can be located at [13].

II. Related Work

Preemptive resource management, a scheme to defend against resource monopolizing DoS, is presented in [8]. This mechanism extensively applies priority and preemption to every type of resource. The concept of malware and botnets working behind DDoS in IoT are discussed in [14], with various DDoS defence techniques compared to identify security gaps and define methods for defending a DDoS attack.

Challenges and mitigation techniques against DDoS attacks in the cloud are surveyed in [2], with a particular focus on using resource quota and fault tolerance at the system level. Available mechanisms to prevent, detect, respond and tolerate DDoS attacks are presented in [12]. It is well known that DDoS attacks are challenging to avert, making it best to maximize the fault tolerance and quality of services. The merits and demerits of each mechanism are discussed, providing a better understanding of the DDoS attack problem.

DDoS attacks, prevention, and mitigation techniques are surveyed in [10]. The most relevant point to our work is load balancing, which involves rerouting data to unaffected network sections by ensuring room for bandwidth allocation. They also consider how to discover an attack as it is heppening, to mitigate the damage of the DoS/DDoS attack, although they do not consider structuring the network to minimize the overload.

A framework to benchmark the throughput of network topologies is presented in [6], then applied to reveal insights about network structure. They show that despite being commonly used, cut-based metrics are ineffective, hence they measure flow-based throughput directly and evaluate topologies with nearly-worst-case traffic matrices (TMs). The framework to benchmark throughput of network topologies presented in [7] uses a two-pronged approach, the first studying the performance of synthetic and experimentally measured TMs, and the other showing how to measure worst-case throughput by generating a near-worst-case TM for any topology.

The method in [9] provides an improved packet delivery ratio by making the network more energy efficient. Multiple metrics were considered, including network lifetime, energy consumption, number of live nodes, and packet delivery rate. Energy consumption characteristics of data transmission over wi-fi are also investigated in [15], focusing on its effect on internet flow characteristics and network environment.

III. Methodology

Our approach uses a genetic algorithm (GA) to evolve networks that are evaluated using three different fitness functions. While the locations of all nodes are preset and do not change over time, the GA determines how to connect those nodes.

A. Topologies

Topologies utilized in this work are abstract network layouts represented in a matrix, where the first row of the matrix represents locations of edge nodes, which transmit the initial requests/data to off-site servers. The last row of the matrix represents the locations of cloud nodes, which are the off-site server nodes receiving the request/data. The final node type is cell nodes, which represent cell towers that receive requests from an edge node and pass it through the network to a cloud node. These are placed in the remaining rows. For each topology, the locations of the nodes are read in during each run of the GA. In this work, the locations were assigned randomly using a separate program, with specified numbers of edge, cloud, and cell nodes.

All topologies examined have 1 edge node, 1 cloud node, and 30 cell nodes, distributed over a 10x10 matrix. The locations of the nodes in topology T0 are shown in Figure 1. We note that the purpose of this study is to evaluate the methodology for allocation of connections between nodes, according to the chosen fitness function, and that any grid dimensions or numbers of nodes can be used. Having used the GA to design the best connections for a given configuration of nodes, a following step is to consider how to handle cybersecurity issues such as when a given node or connection becomes unavailable due to an attack or other problem. The resilience of the network to such issues can be evaluated, and the same methodology used to rebuild or a different one to handle minor adjustments.

B. Self-Driving Automata (SDAs)

The connections between nodes are represented by an adjacency matrix A, where $A[i][j] = 1$ indicates that there is a direct connection between nodes i and j, and 0 otherwise. The entries of the adjacency matrix are filled using the output of *self-driving automata* (SDA).

SDAs, first introduced in [11], are *finite state machines* [5] that have been modified such that each transition is paired with a response of 1 or 2 bits. Further, when a transition is traversed the SDA outputs the response. The output is also appended to an input buffer, which is consumed bit-by-bit to select the next transition. In this way SDAs generate output that is an infinite string of bits with a complex pattern. The SDAs in this paper are initialized to always start at the state with index 0, and store an initial bit used to select the first state transition and be the first character of output. Each state has two transitions emanating from it, corresponding to the bit values of the binary alphabet. The initial bit, destination of transitions, and the size and contents of the response vectors are randomly initialized at the start of evolution. Examples of the use of SDAs to generate networks include [1, 3]. Although these examples were for a different application area, the overall process of using SDA output to fill an adjacency matrix for a network is the same.

Edge				x						
						x		x		
		x	x			x	x	x		
			x					x		x
	x		x			x	x		x	
		x		x		x	x	x		
				x			x	x		x
	x			x	x				x	x
										x
Cloud							x			

Figure 1: Locations (x) of Nodes in Topology T0

C. Genetic Algorithm – Representation

We use a GA to evolve a population of SDAs, with each SDA being used to fill the entries of an adjacency matrix that represents the network connections for a given topology. The adjacency matrix is filled using the SDA output, loading the lower bottom triangle with the SDA output and then assigning the corresponding value on the upper right triangle to that same value such that $A[i][j] = A[j][i]$.

The networks produced by the SDAs are evaluated using the fitness functions described in Section III-D. All of these would give very good fitness scores to networks in which most nodes are directly connected to most others. As a result, without restriction the GA would favour SDAs that produce (adjacency matrices for) such highly-connected networks. However, this is not reasonable in real-world situations in which there is a cost to every connection. As a result, we use a *necrotic filter*, the purpose of which is to kill members of the population if they violate a given set of criteria. In this work, we use a necrotic filter that imposes a limit on the total number of connections, chosen here to be 1–7 times the number of nodes in the network. This is simply a program parameter that was chosen to be similar to related work [1][3], so that we can evaluate the methodology for this problem. However, it can be set to any reasonable value according to requirements from the problem domain.

D. Fitness Functions

Three fitness functions are considered: distance, throughput, and energy consumption. All fitness functions are linked to the desire to reduce latency and so should be *minimized*.

1) Distance: The purpose of the *distance* fitness function is to reward networks with paths from edge nodes to cloud nodes that are as short as possible. It first calculates the Euclidean distance between connected nodes based on their position in the matrix, with each entry in the matrix being considered a unit square in which the node is at its centre. Note that the unit square in question could be, for example, a city block or any other square unit of measurement. It uses those distances to determine the shortest path from each edge node to all cloud nodes using Dijkstra's algorithm. It then averages these distances by (i) for each edge node, adding the lengths of these shortest paths, then dividing by the number of cloud nodes to which it has a path, then (ii) averages again by dividing the summation of the result(s) from (i) by the number of edge nodes. This is then returned as the fitness for the network.

2) Throughput: The second fitness function is *throughput*. This operates by calculating the amount of data being passed through each node in the network to reach the cloud layer of nodes. Its purpose is to prevent the nodes from being overloaded.

It uses a simulation in which multiple streams of data flow from the edge node(s) into the rest of the network. It first layers the nodes in a breadth-first manner, in terms of the number of connections from each of them to a cloud node. When data is passed through the network it can only go from node a to node b if there is a connection between them and b is in a lower layer than a (i.e. closer to the cloud node). It uses a greedy approach that passes data to the node that is receiving the least amount of data of all such nodes.

In the simulation, each edge node generates a queue of packets of random size that total no more than a given maximum value. The edge nodes start sending these packets at regular intervals, until all their data has been sent. Once all data in the network has reached a cloud node the average throughput (average packet size of each cell node divided by number of cell nodes) is returned as the fitness value.

3) Energy Consumption: The final fitness function is *energy consumption*. It uses the same simulation as in throughput fitness. It calculates the average amount of energy consumed in the network by finding, for each node, the total amount of data passed through that node and the (Euclidean) distance each packet travels to the next node in its path. It then multiplies this total by a fixed rate that represents the energy cost to send a megabyte of data over one distance unit (i.e. one cell in the matrix). This is then returned as the fitness value.

E. Experiments

The GA's ability to produce DoS/DDoS resilient network connection configurations is tested with a large number of hyper-parameter settings, dictating the population size, number of mating events, crossover and mutation rate. These experiments are summarized in Tables I and II. All combinations of experiments are considered for all fitness functions, with 30 replicates of each. In all cases, we use size-3 tournament selection and two-point crossover.

Crossover	Mutation	Experiments
10%	10%	E1, E10, E19, E28, E37, E46
10%	50%	E2, E11, E20, E29, E38, E47
10%	90%	E3, E12, E21, E30, E39, E48
50%	10%	E4, E13, E22, E31, E40, E49
50%	50%	E5, E14, E23, E32, E41, E50
50%	90%	E6, E15, E24, E33, E42, E51
90%	10%	E7, E16, E25, E34, E43, E52
90%	50%	E8, E17, E26, E35, E44, E53
90%	90%	E9, E18, E27, E36, E45, E55

Table I: Groupings of Experiments: Crossover and Mutation

Population size	Mating Events	Experiments
50	90,000	E1–E9
50	9,000	E10–E18
100	90,000	E19–E27
100	9,000	E28–E36
250	90,000	E37–E45
250	9,000	E46–E54

Table II: Groupings of Experiments: Population Size and Mating Events

IV. Results

A. Distance

Our examination of the distance fitness function revealed no significant differences in results between the different experiments, with all experiments achieving very similar results. Thus, we omit it from further examination in this paper and will focus on the others, namely throughput and energy consumption.

B. Throughput

Figure 2 presents box plots of the best fitness obtained in all runs for all experiments, for the topology shown in Figure 1 using the throughput fitness function. Several points of interest in the results are visible, the first being that the experiments using 90,000 mating events produce results significantly better than those with only 9,000, showing that the GA requires the additional time to evolve better solutions.

The next noticeable trend is that the solutions using population sizes of 100 and 250 produce slightly better solutions than those using a population size of 50, however this trend is less noticeable when more mating events are used. The overall performance for population sizes of 100 and 250 are nearly identical, showing that although the GA did benefit from increasing the population size, continuously increasing it did not provide a steady improvement to the solutions.

Finally, it is very noticeable that although the confidence intervals overlap, a higher mutation rate results in better fitness. In comparison, changes in crossover rate (with all other settings unchanged) lead to very similar results.

C. Energy

Figure 3 presents box plots of the best fitness obtained in all runs for all experiments, for the topology shown in Figure 1 using the energy fitness function. These results show the same trends as for throughput, namely that the number of mating

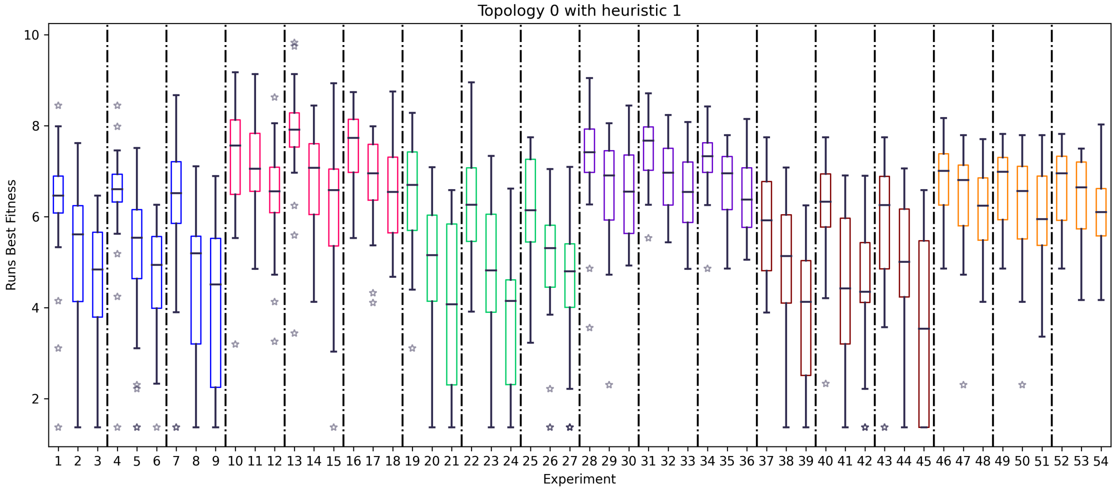

Figure 2: Box-plot distribution for Topology 0 with Fitness Heuristic Measuring Throughput

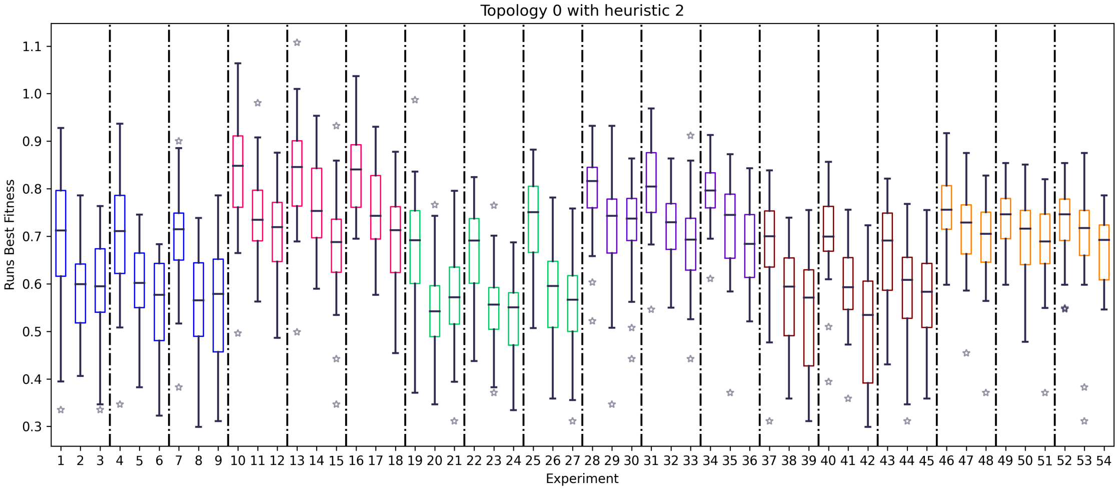

Figure 3: Box-plot distribution for Topology 0 with Fitness Heuristic Measuring Energy Consumption

events and the mutation rate are the hyper-parameters most impacting the fitness of each run.

D. *Focused examination*

The above comparisons of all experiments were performed on only a single topology but identified significant trends. Based on these trends, we selected nine of the best-performing experiments to examine in further detail. These are experiments 3, 6, 9, 21, 24, 27, 39, 42 and 45, as they almost always had the best (i.e. lowest) median fitness for both throughput and energy, as shown in Figures 2 and 3. The hyper-parameters these experiments have in common are mutation rate of 90% and number of mating events equal to 90,000. For this focused examination, a set of five topologies were used: T0 (as before), along with T1, T2, T3, and T4. All topologies have the same characteristics as specified in Section III-A.

Figures 4 and 5 show box-plots for throughput fitness and energy fitness respectively, for all five topologies and each of the nine chosen experiments. In both of these figures, it is visible that for all topologies the GA produces similar best fitness values regardless of the experiment, indicating that these values are not random occurrences and are being produced by the GA. However, the experiment with the best median fitness varies depending on the topology.

We next further examine two of the best-performing experiments across all topologies, in particular experiment 21 (throughput) and experiment 45 (energy consumption). Figures 6 and 7 present the average best fitness for each topology for these experiments, across all 30 runs. These results show the GA improving the population fitness at every reporting interval, which in this case is every 10,000 mating events. This

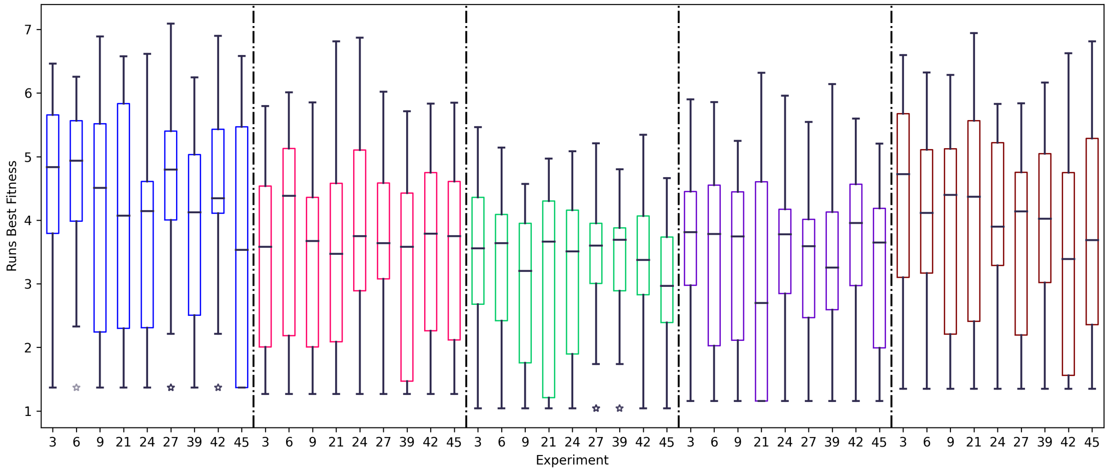

Figure 4: Box-plot for all topologies using throughput fitness for selected experiments. Results for topologies are shown from left to right: T0 (blue), T1 (red), T2 (green), T3 (purple), and T4 (brown).

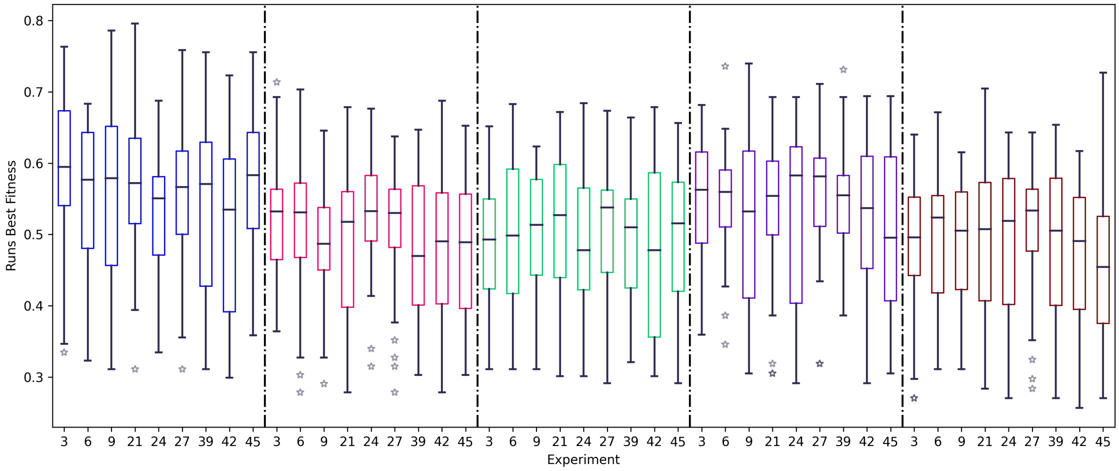

Figure 5: Box-plot for all topologies using energy consumption fitness for selected experiments. Results for topologies are shown from left to right: T0 (blue), T1 (red), T2 (green), T3 (purple), and T4 (brown).

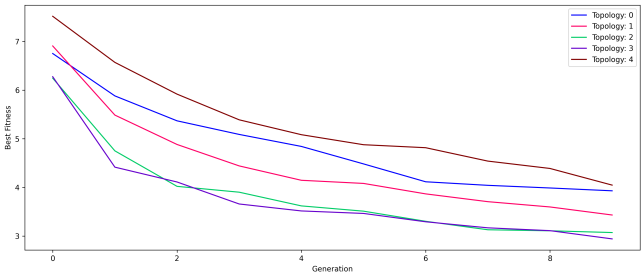

Figure 6: Average best fitness for each topology over 30 runs, using throughput fitness on experiment 21.

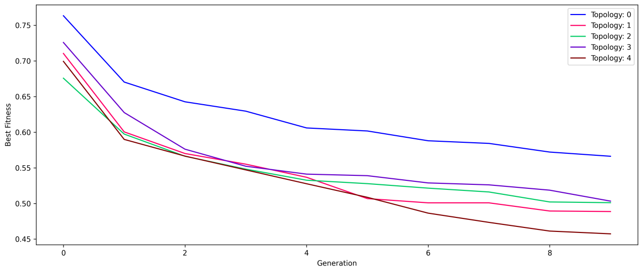

Figure 7: Average best fitness for each topology over 30 runs, using energy fitness on experiment 45.

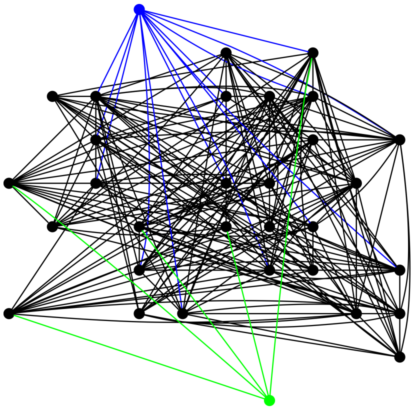

Figure 8: Network with best fitness found for topology T0 (see Figure 1) using throughput fitness on experiment 21.

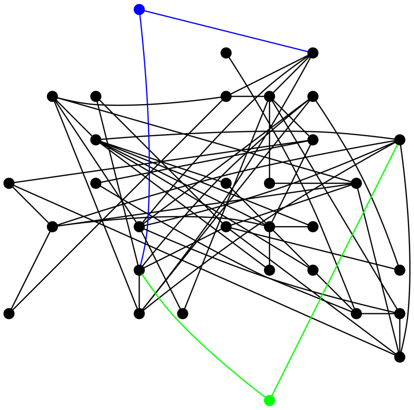

Figure 9: Network with best fitness found for topology T0 (see Figure 1) using energy fitness on experiment 27.

happens for all five topologies, indicating that for any topology the GA should gradually improve the connections relative to the chosen fitness function. The result is that the network will be more tolerant to increases in data transmissions and become less susceptible to DoS/DDoS attacks that operate by overloading the system. A further notable point is that there is no complete plateauing of the lines, indicating additional mating events could still possibly improve fitness.

For the final point of analysis, we examine two of the best networks produced by the GA, using topology T0. These correspond to experiment 21 for throughput fitness and to experiment 27 for energy fitness. These are shown in shown in Figures 8 and 9 respectively. These images reveal both heuristics are choosing connections to satisfy their individual needs. The throughput fitness spreads the transmissions throughout the network, while trying to make connections as direct as possible. Energy fitness is working to choose connections that provide the most direct connection between edge and cloud nodes while distributing data among the nodes. Although these networks had very good fitness, it can be seen that both networks have numerous nodes with connections to distant nodes. This is an issue for communications as it increases the cost and risk of damage to transmissions, indicating a need to fine-tune the heuristics or use a multi-objective approach.

V. Conclusions and Future Work

This paper serves as a proof of concept for employing a GA using SDAs as its representation to design network connection layouts that provide increased resilience against DoS/DDoS attacks. It shows that the GA produces network connection layouts that gradually provide better fitness (throughput or energy consumption) throughout the mating events. This results in throughput or energy consumption being better managed, making them more tolerant of changes in the number of packets transmitted through the network. Thus, networks will become more resistant/tolerant to DoS/DDoS attacks. Another feature of GAs that makes them advantageous in generating different topologies is the fact that they are population based: at the end of evolution the GA produces multiple possible configurations of connections for the same configuration of nodes, with all having similar fitness. In the case that the configuration deployed initially is no longer capable of handling the data flowing through due to a DoD/DDoS attack or node/connection failure, there are alternative configurations to use. In other words, the algorithm has the potential to design connection layouts resilient to DOS/DDoS attacks, along with backup network configurations, to handle scenarios where it becomes overpowered or a connection/node failure occurs.

A. Future Work

Future work will examine restrictions on the number of connections for each node. In this paper, we set that limit to 7 using a necrotic filter (see Section III-C). However, preliminary examinations not included here due to time and space constraints revealed that further limiting the connection count to 5 can improve fitness. Further SDA-related settings should also be examined. For example, the types of allowed responses can be set to favour the generation of approximately the desired number of connections, thereby reducing the work required by the necrotic filter. Another important avenue is to further analyze the best networks created by the GA for each fitness function, which can help to inform possible improvements to them. It is also worthwhile to consider the use of a multi-objective approach, allowing for tradeoffs between the different fitness functions.

Further research will focus on simulating DoD or DDoS attacks that "remove" nodes or edges. Rebuilding could be performed by again using a GA using SDAs, or by other GA-based approaches such as [4]. We will also investigate more extensive networks or topologies that use a larger grid or fewer nodes, resulting in more sparse configurations using fewer edges. Limitations on the distance fitness function will be incorporated to minimize the total lengths of all edges, determining how this would affect the fitness and connections in the network. Other avenues of investigation include examining other fitness functions that provide a new way to define how connections become established in the network.

References

[1] D. Ashlock and M. Dubé. A comparison of novel representations for evolving epidemic networks. In *2021 IEEE CIBCB*, pages 1–8, 2021. doi: 10.1109/CIBCB49929.2021.9562847.

[2] A. Bakr, A. E.-A. Ahmed, and H. Hefny. A survey on mitigation techniques against ddos attacks on cloud computing architecture. *Journal of Advanced Science*, 28:187–200, 11 2019.

[3] M. Dubé and S. Houghten. Now I know my alpha, beta, gammas: Variants in an epidemic scheme. In *2022 IEEE Congress on Evolutionary Computation*, pages 1–8, 2022. doi: 10.1109/CEC55065.2022.9870391.

[4] M. Dubé, S. Houghten, and D. Ashlock. Representation for evolution of epidemic models. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pages 2370–2377, 2019. doi: 10.1109/CEC.2019.8790265.

[5] J. Hopcroft. *Introduction to Automata Theory, Languages, and Computation*. Always Learning. Pearson Education, 2008. ISBN 9788131720479.

[6] S. A. Jyothi, A. Singla, P. B. Godfrey, and A. Kolla. Measuring throughput of data center network topologies. In *The 2014 ACM International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '14, page 597–598, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450327893. doi: 10.1145/2591971.2592040. URL https://doi-org.proxy.library.brocku.ca/10.1145/2591971.2592040.

[7] S. A. Jyothi, A. Singla, P. B. Godfrey, and A. Kolla. Measuring and understanding throughput of network topologies. In *SC16: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 761–772. IEEE, 2016. ISBN 9781467388153.

[8] W. Kaneko, K. Kono, and K. Shimizu. Preemptive resource management: Defending against resource monopolizing dos. In *21st IASTED International Multi-Conference on Applied Informatics*, IASTED International Multi-Conference on Applied Informatics, pages 662–669, 2003. ISBN 0889863415.

[9] Y. Liu, Y. Wang, F. Lombardi, and J. Han. An energy-efficient online-learning stochastic computational deep belief network. *IEEE journal on emerging and selected topics in circuits and systems*, 8(3):454–465, 2018. ISSN 2156-3357.

[10] T. Mahjabin, Y. Xiao, G. Sun, and W. Jiang. A survey of distributed denial-of-service attack, prevention, and mitigation techniques. *International Journal of Distributed Sensor Networks*, 13(12):1550147717741463, 2017. doi: 10.1177/1550147717741463.

[11] A. McEachern and D. Ashlock. Shape control of side effect machines for dna classification. In *2014 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology*, pages 1–8, 2014.

[12] A. Mishra, B. Gupta, and R. Joshi. A comparative study of distributed denial of service attacks, intrusion tolerance and mitigation techniques. In *2011 European Intelligence and Security Informatics Conference*, pages 286–289, 2011. doi: 10.1109/EISIC.2011.15.

[13] K. Olenic. Building paths to reduce latency and increase resilience to cyberattacks - ssci 2025. URL https://github.com/ko19af/GA-s-and-SDA-s-SSCI-2025.git.

[14] R. Vishwakarma and A. Jain. A survey of ddos attacking techniques and defense mechanisms in the iot network. *Telecommunication Systems*, 73, 01 2020. doi: 10.1007/s11235-019-00599-z.

[15] Y. Xiao, Y. Cui, P. Savolainen, M. Siekkinen, A. Wang, L. Yang, A. Ylä-Jääski, and S. Tarkoma. Modeling energy consumption of data transmission over wi-fi. *Mobile Computing, IEEE Transactions on*, 13:1760–1773, 08 2014. doi: 10.1109/TMC.2013.51.