Forging Paths for DoS/DDoS Resilient Networks

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

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

***Abstract*—In today's technology focused environment, networks receive a constant multitude of requests from various user devices, leading to cases of high latency in the network if data transmissions are mismanaged. If a denial of service (DoS) or distributed denial of service (DDoS) attack is launched on a network with mismanaged data transmissions, the damage would be catastrophic. Thus, appropriate management of transmissions is paramount. Methods for managing communications exist, but they require costly computations and account for select variables such as energy consumption or the amount of data transmitted. Our approach utilizes genetic algorithms to generate network configurations with structures that reduce latency based on different factors: data transmitted over a connection, and energy consumption. Once designed, their resilience to DoS/DDoS attacks is evaluated. The results show the methodology designs networks resistant to attack and that the surviving networks can also be reconfigured to re-establish the fitness to a pre-attacked state.**

I. Introduction

The aim of *Denial of Service* (DoS) and *Distributed Denial of Service* (DDoS) attacks is to disrupt the normal operation of a targeted system or network by overwhelming it with traffic, rendering it unavailable to legitimate users. In *volumetric* DDoS attacks a malicious individual attempts to overwhelm a server with an immense volume of requests, exhausting its bandwidth and processing power. The difficulty with these attacks is that they leverage multiple compromised systems (such as bots) to flood the target with traffic, making it difficult to identify the malicious traffic. Also, once an attack occurs it is crucial to reconfigure the network as quickly as possible to maintain network operations.

In [15], network connection layouts were designed using a *genetic algorithm* (GA). With the goal of minimizing network latency, three objectives were independently considered: transmission distance, throughput, and energy consumption. In so doing, the overall aim was to build networks that were resilient to DoS and DDoS attacks. However, the effect of such attacks was not directly measured.

Contributions: In this paper, we use a similar process to design networks. We then examine their resilience against volumetric attacks by performing simulations on them to mimic their effects: disabling of nodes or pushing additional data onto the network. Further, our process restructures networks to compensate for the attack when required. Also, because the GA produces a population of viable solutions (networks) these alternative configurations, that account for the damage caused by the attack, can be used should another attack occur. In fact, we show that the majority of these networks can so be used.

The code for this research is available at: github.com/ko-19-af/GA-s-and-SDA-s-WCCI-2026.

II. Related Work

A scheme to defend against resource-monopolizing DoS is presented in [7], using a mechanism that extensively applies priority and preemption to every type of resource. In [16], techniques to identify security gaps and defend against DDoS attacks are compared, along with the concept of malware and botnets working behind DDoS in IoT. Challenges and mitigation techniques against DDoS attacks in the cloud are surveyed in [2], particularly methods using resource quota and fault tolerance at the system level. Mechanisms for preventing, detecting, tolerating, and responding to DDoS attacks are presented in [12], focusing on maximizing fault tolerance and quality of services. DDoS attacks, prevention, and mitigation techniques are surveyed in [10], including load balancing — rerouting data to unaffected network sections, ensuring room for bandwidth allocation.

A framework to measure the throughput of network topologies, revealing information on network structure, is presented in [5]. The authors measure flow-based throughput directly and evaluate topologies with nearly-worst-case traffic matrices, showing cut-based metrics are ineffective. A method to improve the packet delivery ratio by making the network more energy efficient is provided in [9]. Metrics considered are network lifetime, energy consumption, number of live nodes, and rate of packet delivery. Energy consumption characteristics of data transmission over wi-fi are investigated in [17], in particular the effect on internet flow and network environment.

There are several recent examples of the use of computational intelligence for related problems. An approach to identify malicious packets in network communications is described in [6], comparing particle-swarm optimization and GAs. This demonstrates how GAs can effectively detect and mitigate traffic diversion attacks in software-defined networking. A hybrid Grasshopper optimization algorithm and GA, in conjunction with the random forest feature selection method, is used to prevent and detect DDoS attacks in [13]. An approach to design networks to reduce latency and increase their resilience to DoS/DDoS attacks is presented in [15]. As in the current work, GAs are used to evolve the networks, with Self-Driving Automata (see Section III-B) as the representation in the GA.

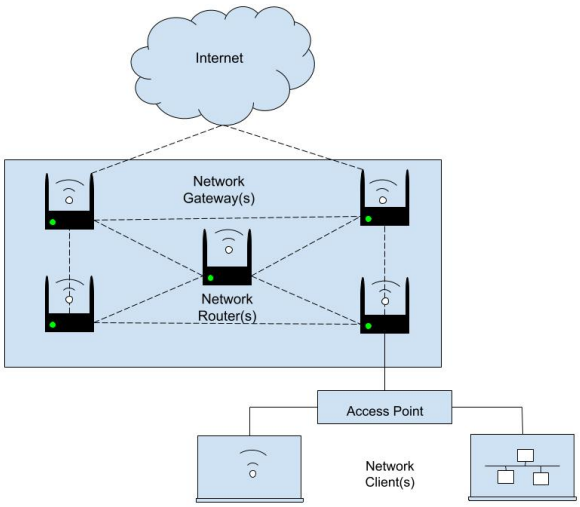


Figure 1: Mesh network representation of a layout and its associated connections, where the routers are cell nodes, clients are edge nodes, and the cloud node is the internet.

Edge node				E					
						X		X		
		X	X			X	X	X		
			X					X		X
Cell nodes	X		X			X	X		X	
		X		X		X	X	X		
				X			X	X		X
	X			X	X				X	X
										X
Cloud node							C			

Figure 2: Locations of nodes in layout T0.

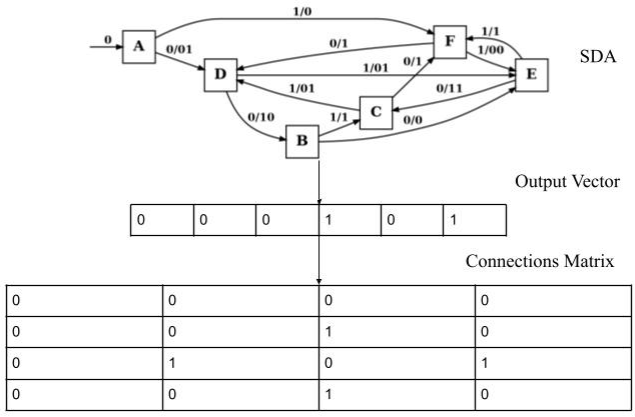


Figure 3: Example of an SDA producing a vector with a two letter alphabet. The output is used to fill the lower triangle of the matrix in row-major order and copied to the upper triangle.

III. Methodology

A. Networks and Layouts

Networks consist of three types of nodes: *edge nodes* transmit initial requests/data to off-site servers, *cloud nodes* receive the requests/data, and *cell nodes* represent routers that receive requests from edge node(s) and pass them through the network to a cloud node. Edge nodes, cloud nodes, and cell nodes correspond to the network clients, network gateways, and network routers, respectively, in Figure 1.

In our approach, we view a network as a graph in which each node corresponds to a router and each edge to a connection between them. As such, throughout this work we use the terms *node* and *router* interchangeably, as well as the terms *edge* and *connection*. In these networks, the locations of the nodes are preset according to a given *layout* and the GA, which evolves the networks, determines how to connect the nodes. To analyze the algorithm and parameters, we examine five layouts with the same basic characteristics: a 10×10 grid with 1 edge node in the first row, 1 cloud node in the last row, and 30 cell nodes distributed across the remaining 8 rows. The locations of the nodes in layout T0 are shown in Figure 2, while the others (T1, T2, T3, and T4) are given in the appendix of [14].

B. Self Driving Automata (SDAs)

Each member of the population in our GA corresponds to a network. In particular, each network is represented by a self-driving automaton (SDA) [11]. These are modified finite state machines, inspired by *Kolakoski sequences* [8]. This representation allows for strong exploration of the space of potential networks, and has been used with great success in a similar manner for a variety of applications, including [1, 3, 15].

In an SDA, each transition between states is paired with a response. The response is sent as output, which drives future transitions. Figure 3 shows a simple SDA with a binary alphabet, with each response (shown after / in each transition) having 1-2 bits. In this figure, the initial state is A and the initial symbol (0) is output and also causes the transition to state D. The response 01 is then sent to output, and the most recent bit (1) causes a transition to state C while outputting 01, and so on. The output can be used to fill any data structure. In this example it is used to fill an adjacency matrix (connection matrix) for a graph, in the order specified in the figure. We note the minor detail that in some other work, transitions are triggered by the first still-unused bit in the output rather than the most recent bit.

C. Genetic Algorithm Specifics

The GA's ability to produce DoS/DDoS resilient networks is tested with settings, dictating crossover and mutation rates, population size, and number of mating events. These are summarized in Tables I and II. All experiments use two-point crossover, random resetting mutation, elitism set to 2, and tournament selection with tournament size set to 3. As in [15], we also cap the total number of edges to 7 times the number of nodes, i.e. $7 \times 32 = 224$. All combinations of settings in Tables I and II are used, with 30 replicates for each experiment.

Two fitness functions are considered: *throughput* and *energy consumption*. The fitness functions are linked to the goal of

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, E54

Table I: Experiments: Crossover and Mutation

Population size	Mating Events	Experiments
50	100,000	E1–E9
50	10,000	E10–E18
100	100,000	E19–E27
100	10,000	E28–E36
250	100,000	E37–E45
250	10,000	E46–E54

Table II: Experiments: Population Size and Mating Events

reducing latency and so should be *minimized*. For both of these, the fitness of a given network is calculated by running a simulation where each edge node generates a queue of packets of random size totaling no more than a given maximum value. The edge nodes send these packets at regular intervals through cell nodes, until all the data reaches a cloud node.

1) Throughput: We modify the throughput function from [15] to improve the networks produced. In particular, our function performs the above simulation and calculates the fitness as the average packet size of each cell node divided by the number of *active* cell nodes (as opposed to *all* cell nodes) in the network. Looking only at active cell nodes encourages the algorithm to utilize more cell nodes, i.e. the data each router must process is spread out and lowered.

2) Energy Consumption: Using the above simulation, the heuristic calculates, for every connection, the total data transmitted through it and its length (Euclidean distance between the nodes it connects). It then multiplies these two values by a fixed rate that represents the energy cost of sending 1 megabyte of data over 1 distance unit (1 cell in the matrix). Finally this is averaged by the number of nodes that consume energy, excluding the edge nodes, and returned as the fitness value.

D. Network Attacks

Two types of attack are considered:

Router shutdown: the attack removes some of the routers in the network. This is simulated by removing all connections to these nodes, indicating that they no longer receive data due to being overloaded.

Additional data: the attack pushes additional data onto the network in an attempt to increase its latency. This is simulated by an intermediate node injecting enough data to increase overall load but not enough to cause a router failure.

For each of these, a specified percentage of routers are affected by the attack. This study examines the effect of 20% of the cell nodes suffering from the attack. An examination of 10% was also performed but the results were similar to those of 20%, so they are excluded from this work. Note that these attacks are not applied to edge or cloud nodes, as disabling those nodes would result in a complete failure of the network.

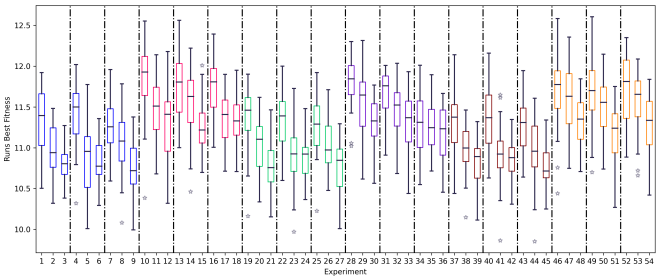


Figure 4: All experiments for layout T0 with throughput fitness and an SDA maximum response length of 2.

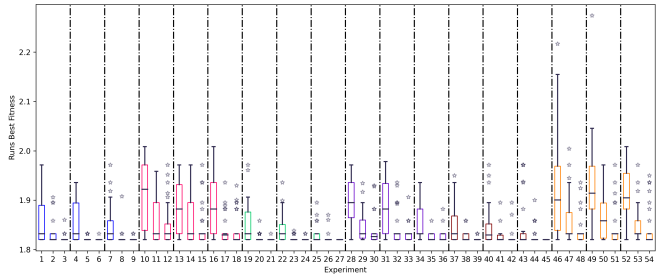


Figure 5: All experiments for layout T0 with energy consumption fitness and an SDA maximum response length of 2.

IV. Results

A. Creation of Initial Networks

We first use the GA to design the initial networks, which will later be examined under attack conditions in Section IV-B.

In [15], an SDA with a maximum response length of 496 characters per transition was used, as this is the maximum number of edges for a 32 node graph. In this paper, SDAs that permit a maximum response length of only 2 characters are used instead, similar to other work such as [3].

Figures 4 and 5 present the box plots of the best fitness obtained in all runs for all experiments for layout T0 (shown in Figure 2) using the throughput and energy consumption fitness functions respectively. Compared to the results in [15], there is notable improvement in the best fitness value and a tightening of the box-plot distribution. This is likely tied to the fact that when the SDA has a maximum response length of 2 characters, it is easier to modify the output vector because any modifications to a transition or response affect the output of the SDA. In contrast, when the output length is 496 characters, the algorithm designs SDAs where if one state outputs the maximum response length then it only needs to alter that response to affect the output.

For layout T0, the best-performing experiments for both fitness functions are experiments 3, 6, 9, 21, 24, 27, 39, 42, and 45. We now focus on these experiments and apply them to all five layouts, again using the 2 character response length.

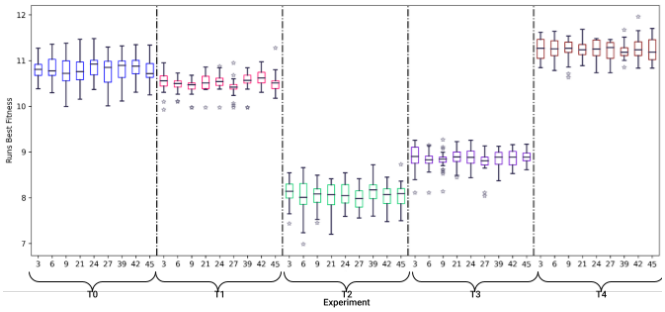


Figure 6: Layouts T0–T4 for throughput fitness and an SDA maximum response length of 2.

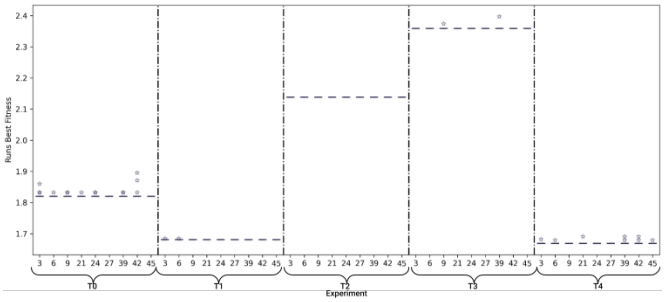


Figure 7: Layouts T0–T4 for energy consumption fitness and an SDA maximum response length of 2.

Figures 6 and 7 show the box-plots for throughput and energy consumption fitness, respectively. In comparison to [15], that used the maximum response length of 496, our results have lower (i.e. better) fitness values for all five layouts. This indicates that the algorithm is designing SDAs that produce networks that better satisfy each heuristic with a near-optimal configuration.

Next, we examine the best-performing experiments for all layouts, namely experiments 9 (throughput) and 27 (energy consumption). Figures 8 and 9 present the average best fitness for each layout for these experiments over 30 runs. The results show that the GA is rapidly improving the fitness of the SDAs as each line nearly plateaus after the first few generations for each layout. In comparison to [15], our results have better fitness and plateau faster.

For the final point of analysis of the initial networks, we examine the best networks produced by the algorithm using layout T0. The networks examined all correspond to experiment 27. These are shown in Figures 10 and 11 for throughput and energy consumption fitness, respectively. These figures reveal that the algorithm chooses connections satisfying the heuristics. For throughput this means generating more connections to spread out the transmissions for each connection, while for energy consumption it means reducing the number of connections to reduce the energy consumption of each node.

B. Attacked Networks

We now use the GA to evaluate the networks under the two types of attack. The router shutdown attack is studied

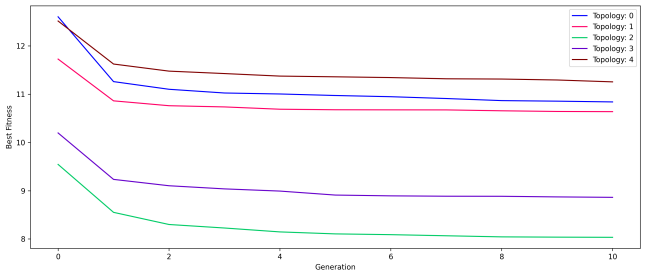


Figure 8: Average best fitness for all layouts over 30 runs for throughput fitness and response length 2 on experiment 6.

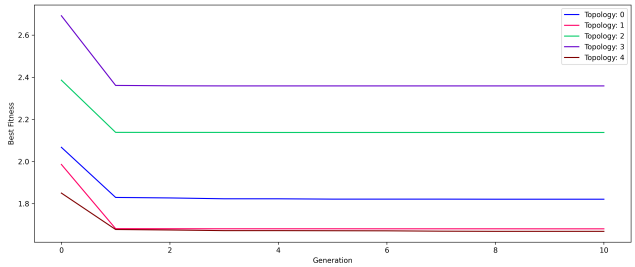


Figure 9: Average best fitness for all layouts over 30 runs for energy fitness and response length 2 on experiment 27.

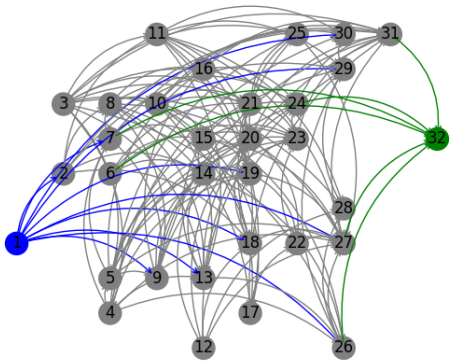


Figure 10: Network with best fitness for layout T0 using throughput fitness and response length of 2 on experiment 27.

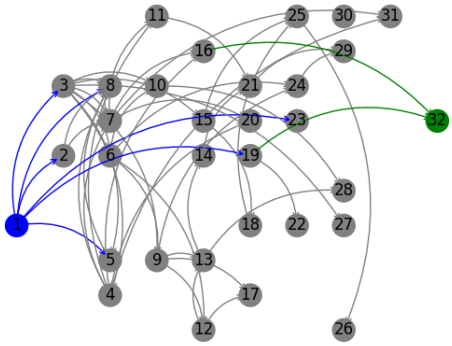


Figure 11: Network with best fitness for layout T0 using energy fitness and response length of 2 on experiment 27.

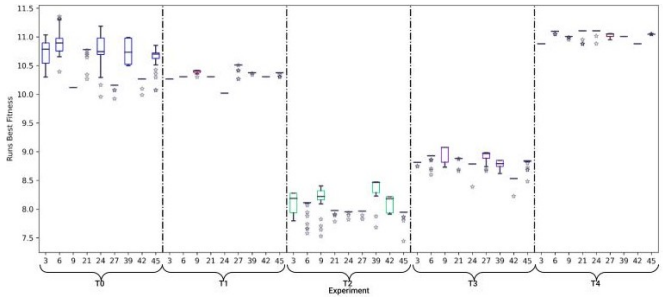


Figure 12: Layouts T0–T4 with throughput fitness after the network is attacked with a 20% router shutdown for experiments 3, 6, 9, 21, 24, 27, 39, 42 and 45.

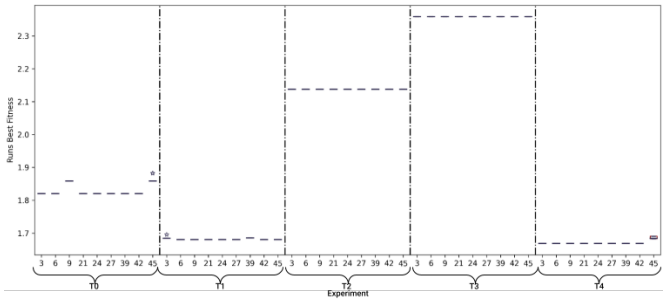


Figure 13: Layouts T0–T4 with energy consumption fitness after the network is attacked with a 20% router shutdown for experiments 3, 6, 9, 21, 24, 27, 39, 42 and 45.

in Section IV-B1 and the additional data attack is studied in Section IV-B2.

1) Router Shutdown Attack: We first performed a detailed examination using all experiments for layout T0, then selected the nine best-performing experiments for analysis on all five layouts. These are experiments 3, 6, 9, 21, 24, 27, 39, 42, and 45 for both fitness functions.

Figure 12 presents boxplots for all five layouts using the throughput fitness function after an attack disabling 20% of the network routers. The boxplots show that the results did not worsen compared to those in Figure 6, which correspond to the network prior to the attack, but rather improved and converged. This indicates that the SDAs withstood the attack, allowing the algorithm to further evolve the solution space. In cases where the networks survived, the algorithm took those SDAs and proceeded as if it were extending the run. However, when the networks became damaged, the algorithm instead restructured the solution space using the original network as a base to mitigate the damage. Thus, surviving networks reduced the time required to reconstruct the network, allowing the algorithm to develop new networks possessing a fitness equal or similar to the original.

Further examination also shows some of the experiments improving, even after the attack. This is not because the attack improves the network. Rather, the initial solutions have not reached their local minima. As a result of them surviving the attack, they will continue to improve until they converge.

Figure 13 presents boxplots for all five layouts using the energy consumption fitness function after an attack disables 20% of the routers in the network. These experiments show the results did not worsen significantly compared to the results in Figure 7, which shows the experiments before the attack. Instead they improved and converged, as in Figure 12.

Figures 12 and 13 demonstrate that these results occur regardless of the layout and experiment, indicating the values do not occur randomly. However, the experiment with the best median fitness varies depending on the layout.

We next further examine the best-performing experiments across all layouts, in particular experiment 21 (throughput) and experiment 6 (energy consumption). Due to space constraints, the convergence plots are excluded here, but they are essentially the same as Figures 8 and 9. The results showed that in most cases the fitness of the experiments plateau, while others improve their fitness once and then plateau. This provides further evidence that the initially designed networks that survived the attack are in local minima. Thus, the network does not improve beyond that level. In cases where the network was damaged but not dead (i.e. no paths connecting an edge node to a cloud node), the algorithm worked to restore the fitness of the network. In addition, since the network survived the attack, it provided a suitable base from which to work, resulting in an expedited restoration process.

Attack %	10%	20%
Throughput	0/30	0/30
Energy Consumption	0/30	13/30

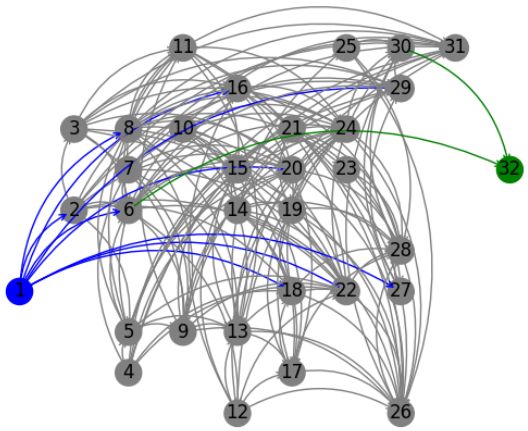
Table III: Number of dead networks from router shutdown attack for each fitness function.

Finally, we examine the best networks produced by the GA, using layout T0, after the router shutdown attack is performed. These are experiment 24 (Figure 14) for throughput and experiment 27 (Figure 15) for energy consumption. These reveal that the heuristics are re-adjusting the connections in the network to compensate for the loss of nodes. The networks presented, which are suffering from the attack, refocus their connections to nodes not affected by the attack, while also increasing the number of connections to those same nodes. This indicates that the algorithm designs networks that are resistant and can perform restructuring to mitigate damage, restoring the networks to a fitness very close to that of the original network.

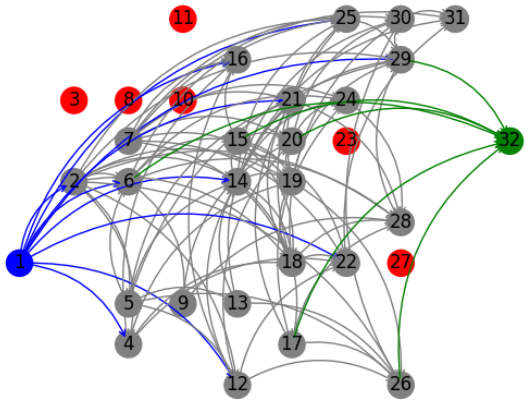
Additionally, Table III shows that the majority of initial networks survived the attack. Thus, when a given configuration fails, another is deployable.

2) Additional Data Attack: We first performed a detailed examination using all experiments for layout T0, then selected the nine best-performing experiments for analysis on all five layouts. For throughput fitness, these experiments are 2, 5, 6, 19, 21, 24, 26, 39, and 41. For energy consumption fitness, these experiments are 3, 6, 9, 21, 24, 27, 39, 42, and 45.

Figure 16 presents box plots of the best fitness obtained in all runs for all experiments, for all five layouts for the

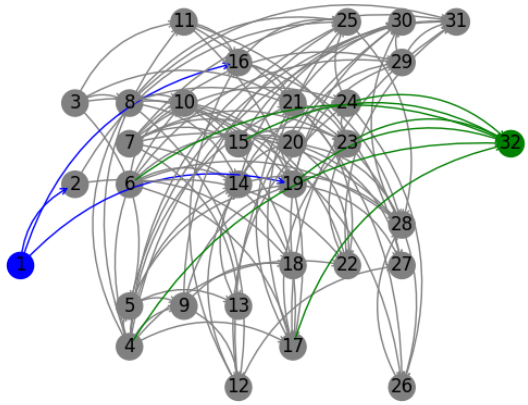


(a) Original

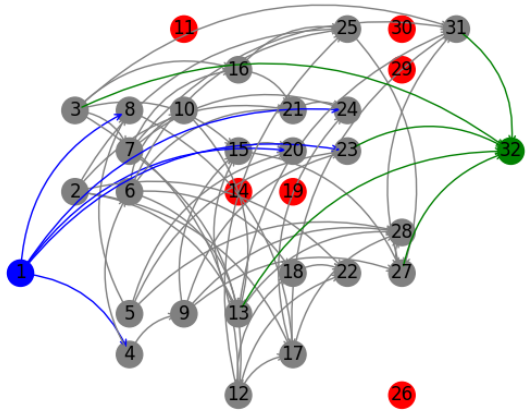


(b) Attacked

Figure 14: Comparison of networks, using throughput fitness for experiment 24, before and after a 20% router outage attack. Red nodes are routers disabled by the attack. Fitness of networks: original = 10.29, attacked = 10.39.



(a) Original



(b) Attacked

Figure 15: Comparison of networks, using energy consumption fitness for experiment 27, before and after a 20% router outage attack. Red nodes are routers disabled by the attack. Fitness of networks: original = 1.82, attacked = 1.86.

throughput fitness function after an attack causing 20% of the routers to transmit additional data is perpetrated on the network. These boxplots show that some experiments were unable to fully acclimate to the increase in data, as the SDAs of the initial design process show varied success. However, this does not indicate that the algorithm is performing worse, but rather that the local minima shifted due to the increase in data. Additionally, some experiments were able to achieve reasonably low (i.e. good) fitness when given the additional data load, while others stagnated with slightly worse fitness. Thus, while the throughput fitness function designs resistant networks, significant rework on some of the network designs may be necessary to further restore the fitness.

Figure 17 shows box plots of the best fitness obtained in all runs for all experiments for all five layouts using the energy consumption fitness function after an additional data attack of 20% occurs. From this figure we see that, unlike the results for throughput fitness (Figure 16), the algorithm was able to reach convergence, as the SDAs of the initial design step deal with the influx of new data entering the network. This indicates that using energy consumption fitness, the GA is capable of developing SDAs that design networks resistant to increases in transmissions within the network.

Additionally, Figures 16 and 17 show that for all layouts, the GA produces similar best fitness values regardless of the experiment, indicating that these values are not random occurrences but are produced by the GA. However, the experiment with the best median fitness varies depending on the layout.

An examination of the best performing experiments was examined in all layouts, in particular experiments 41 and 6 (throughput), and experiments 39 and 27 (energy consumption), attacking 20% of the nodes for each heuristic. Although excluded here due to space constraints, the results were similar to those in Figures 8 and 9, displaying minor improvement followed by immediate plateauing.

For the final point of analysis, we examine the best networks

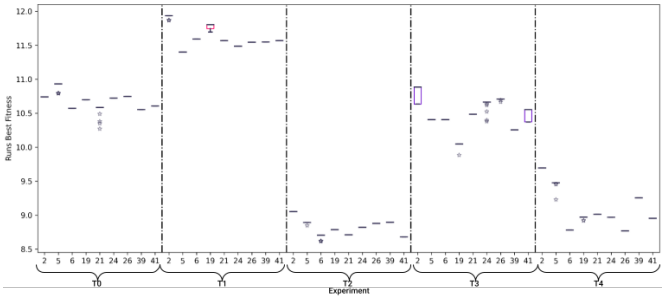


Figure 16: Layouts T0–T4 with throughput fitness after the network is attacked with 20% of the routers receiving additional data for experiments 2, 5, 6, 19, 21, 24, 26, 39, 41.

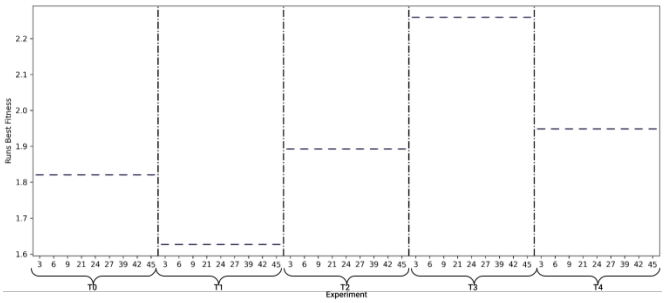


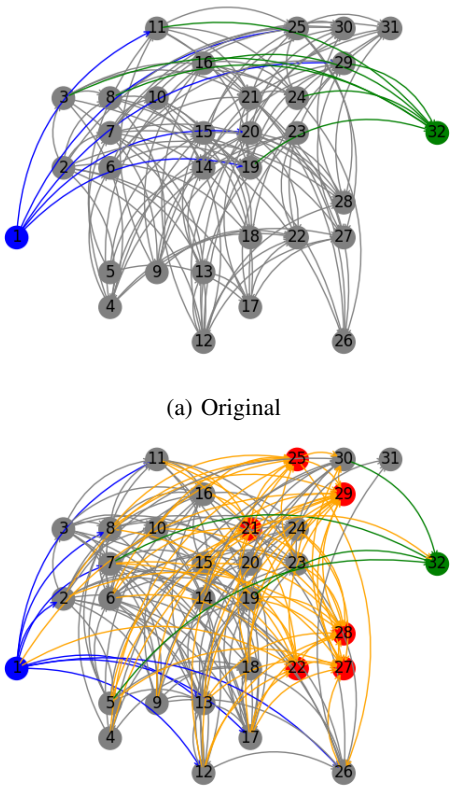
Figure 17: Layouts T0–T4 with energy consumption fitness after the network is attacked with 20% of the routers receiving additional data for experiments 3, 6, 9, 21, 24, 27, 39, 42, 45.

produced by the GA, using layout T0, suffering from an additional data attack. These correspond to experiment 24 for throughput and experiment 27 for energy consumption fitness, and are shown in Figures 18 and 19, respectively. These figures reveal that the algorithm has designed networks that modify the original network connections to reduce the strain placed on the network, restoring the fitness score to a value very similar to that of the network before the attack. This is due to the network isolating the attacked nodes and removing/reducing any direct or indirect connections to the affected nodes.

V. Conclusions and Future Work

Our main contribution in this paper demonstrates genetic algorithms using SDAs as their representation are capable of producing connection layouts with increased resilience against DoS/DDoS attacks.

Additionally, using the SDAs that survived the attacks as a base for the GA's population, the algorithm improved their fitness back to a level similar to pre-attacked networks. If not, then it was at least at an acceptable level based on the circumstances of the attack. However, some networks could not achieve their original fitness, due to paths no longer being available and some of the attacks pushing additional data into the system, preventing it from achieving the same level due to the minima changing. Nonetheless, they achieved a design that dealt with the data based on the heuristic used, similar to the results of [15].



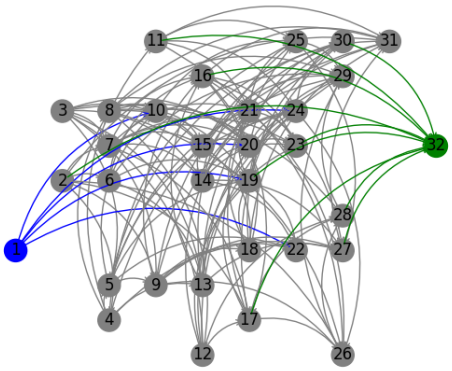
(a) Original

(b) Attacked

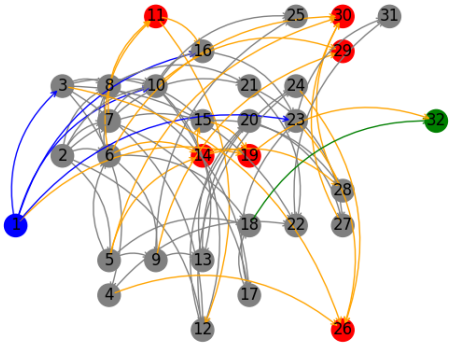
Figure 18: Comparison of networks, using throughput fitness for experiment 6, before and after a 20% additional data attack. Orange nodes are routers affected by the attack. Fitness of networks: original = 10.31, attacked = 10.61.

It was also confirmed that while some networks died (i.e no connections between the edge and cloud node(s) remain), other networks with different configurations survived. Thus, the GA can design alternative networks and when a network is disabled by the attack, an alternative network can be deployed, allowing the network to continue operating.

Future work will examine more extensive layouts that use a larger grid or fewer nodes, resulting in more sparse configurations using fewer edges. This investigation could also look at using the smaller SDAs built for the networks in the current work and applying them to bigger networks. This would determine if the qualities associated with the fitness function are passed on to the bigger networks or if the use of the pre-made SDAs reduces the amount of time needed to produce networks possessing a satisfactory fitness. Other avenues of investigation include the examination of a multi-heuristic approach and other fitness functions, including the Received Signal Strength Indicator (RSSI), Signal-to-Noise Ratio (SNR), Link Quality, etc. Integrating these heuristics into the algorithm and optimizing them would allow network administrators to detect anomalies more easily and potentially identify malicious activity that could be part of a DDoS attack,



(a) Original



(b) Attacked

Figure 19: Comparison of networks, using energy consumption fitness for experiment 27, before and after a 20% additional data attack occurs. Orange nodes are routers affected by the attack. Fitness of networks: original = 10.00, attacked = 10.75.

thus providing a new way to define how connections become established in the network. Additionally, we can alter how the algorithm rebuilds networks when using the genetic algorithm using SDAs, or by other GA-based approaches such as [4].

Acknowledgements

This research was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) and made possible by the facilities of SHARCNET and the Digital Research Alliance of Canada.

References

[1] D. Ashlock and M. Dubé. A comparison of novel representations for evolving epidemic networks. In *2021 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology*, pages 1–8, 2021.

[2] A. Bakr, A. Ahmed, and H. Hefny. A survey on mitigation techniques against ddos attacks on cloud computing architecture. *International Journal of Advanced Science and Technology*, 28(12):187–200, 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.

[4] M. Dubé, S. Houghten, and D. Ashlock. Representation for evolution of epidemic models. In *2019 IEEE Congress on Evolutionary Computation*, pages 2370–2377, 2019.

[5] S. Jyothi et al. Measuring throughput of data center network topologies. In *ACM International Conference on Measurement and modeling of computer systems*, page 597–598, 2014.

[6] K. Kalpana, , M. Prachi, and S. Paranjape. Enhancing network security: A strategic SDN and genetic algorithm approach for countering DDoS attacks. In *International Conference on Information and Communication Technology for Intelligent Systems*, pages 617–627. Springer, 2024.

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

[8] W. Kolakoski. Self generating runs (problem 5304). *American Mathematical Monthly*, 72(674), 1965.

[9] Y. Liu et al. 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.

[10] T. Mahjabin et al. A survey of distributed denial-of-service attack, prevention, and mitigation techniques. *International Journal of Distributed Sensor Networks*, 13 (12):1550147717741463, 2017.

[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 *European Intelligence and Security Informatics Conference*, pages 286–289, 2011.

[13] S. Mohammadi and M. Babagoli. A hybrid modified grasshopper optimization algorithm and genetic algorithm to detect and prevent DDoS attacks. *International Journal of Engineering (Tehran)*, 34(4), 2021.

[14] K. Olenic. Reducing latency and increasing resilience to cyberattacks on networks. *MSc. Thesis, Brock University*, 2025.

[15] K. Olenic, M. Dubé, and S. Houghten. Building paths to reduce latency and increase resilience to cyberattacks. In *IEEE Symposium on Computational Intelligence in Security, Defence and Biometrics*, pages 1–7, 2025.

[16] R. Vishwakarma and A. Jain. A survey of DDoS attacking techniques and defense mechanisms in the IoT network. *Telecommunication Systems*, 73, 01 2020.

[17] Y. Xiao et al. Modeling energy consumption of data transmission over Wi-Fi. *IEEE Transactions on Mobile Computing*, 13:1760–1773, 08 2014.