A Comparison of Neighborhood Topologies in Particle Swarm Optimization

Jigyasa Subedi, Souleman Toure, Diyaa Yaqub

Abstract

We implemented the Particle Swarm Optimization (PSO) algorithm which involves particles within a swarm moving around a search space to look for the best solution. We wanted to assess the role that different neighborhood topologies played in finding the optimal solution.The neighborhood topologies we used were global topology, ring topology, von Neumann topology and random topology. To evaluate how the algorithm performs on these various topologies, we used test functions Ackley, Rastrigin, and Rosenbrock. For each test function, we observed the proximity of the particles to the global minimum for that function. To evaluate the algorithm, we compared the best value found by particles in the solution space at the end of 10,000 iterations for each of the topologies with each test function.

1 Introduction

In this project, our goal was to assess the impact of neighborhood topology on the performance of the Particle Swarm Optimization (PSO) algorithm. The PSO algorithm is a stochastic optimization technique based on swarm behavior. It uses a simple mechanism that mimics swarm behavior in birds flocking and fish schooling to guide the particles to search for global optimal solutions. The neighborhood topologies we used in our algorithm were global, ring, von Neumann and random. We used a series of single objective optimization test functions to evaluate this algorithm. These functions are Rosenbrock, Rastrigin and Ackley. The Rosenbrock function is unimodal, and the global minimum lies in a narrow, parabolic valley. The Rastrigin function has several local minima. It is highly multimodal, but locations of the minima are regularly distributed. The Ackley function in its two-dimensional form, is characterized by a nearly flat outer region, and a large hole at the center. These test functions play an important role in validating and assessing the performance of this algorithm.

Using 36 tests, we analyzed different combinations of particles and topologies across the three functions. We found that generally von Neumann and the random topologies produced the best results. In addition to that, we also found that increasing the swarm size produced better results and had positive effects on the optimization.

The remainder of the paper is organized as follows: Section 2 provides an overview of PSO, the neighborhood topologies and the fitness functions used. Section 3 presents our experimental methodology. Section 4 discusses the results of the performed experiments. Finally, Section 5 and 6 provide the concluding remarks and ideas for future work.

2 Particle Swarm Optimization

PSO is a stochastic optimization technique based on the movement and intelligence of swarms. It was proposed by Eberhart and Kennedy in 1995. PSO uses the concept of social interaction to solve the given problem. It uses a number of particles (candidate solutions) that constitute a swarm moving around in the search space, looking for the best position. Particles will move through a multidimensional search space to find the best position, and thus the best solution, in that space. This algorithm is inspired from swarm behavior such as bird flocking. Particles tend to keep moving in the current direction and same speed, but are biased towards their own experience and the swarm's experience. The goal is to find better and better solutions. In each iteration, each particle is updated using two "best" values: pbest and nbest. Pbest, personal best, is the best solution that particle has found so far. Nbest, the neighborhood best, is the best solution that any particle in the swarm has found so far within the particle's neighborhood. A swarm of particles fly through the search space being attracted by both, their personal best position and the neighborhood best position found so far within the swarm.

The general outline of the algorithm is as follows. Our PSO algorithm works to solve the functions by generating a user-inputted number of particles. Each particle is initialized with a random position and velocity (which are vectors in the d-dimensional space). Each particle's position is used to evaluate the function. The solution it finds is stored but updated as the position and velocity change. Each particle stores and tracks its personal best (the minimum value). The particles are also divided into neighborhoods. The best solution from each of the particles in the neighborhood (the neighborhood best) and the particle's personal best, is used to update each particle's velocity vector. Although the equation is influenced by these values, it also introduces randomness through the generation of random numbers and the constriction factor. According to this updated velocity vector, the position is updated, and the new position is used to find the solution to the functions. This continues for a specified number of iterations and the best solution found between all the particles is outputted.

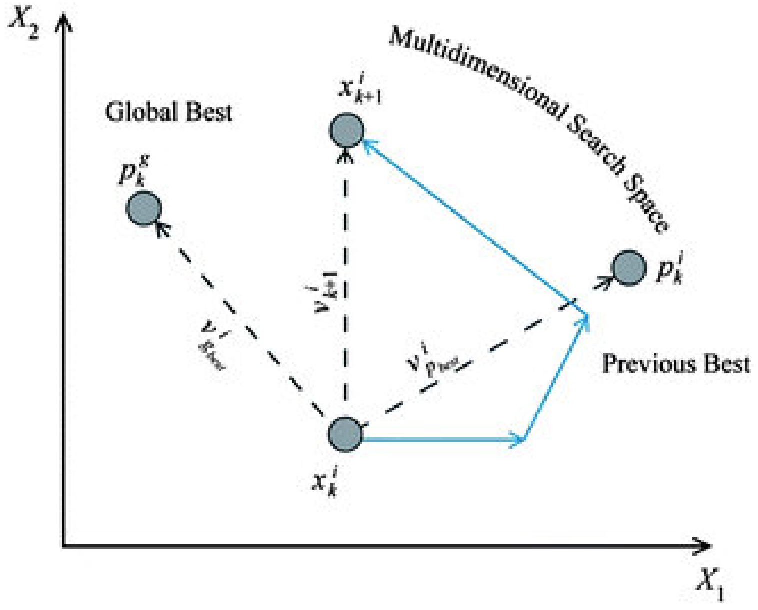


Figure 1: Depiction of the velocity and position update

Just a note, our PSO algorithm uses constriction factor instead of inertia as the constriction factor keeps the velocity in check. The equations used to update the velocity and position are shown below:

$$\vec{v_i} \leftarrow \chi(\vec{v_i} + \vec{u}(0,\phi_1) \otimes (\vec{p_i} - \vec{x_i}) + \vec{u}(0,\phi_2) \otimes (\vec{g_i} - \vec{x_i}))$$
$$\vec{x_i} \leftarrow \vec{x_i} + \vec{v_i}$$

The PSO algorithm has a lot of advantages. It is easy to implement and relatively fast. It can handle dynamic problems quite efficiently. More importantly, compared to an algorithm like the genetic algorithm, it has fewer individuals and fewer parameters with better rule-of-thumb values. It is easy to hybridize with other algorithms to improve its performance. It has also been applied successfully to a wide variety of search and optimization problems. Some PSO applications include neural networks, communication networks, image and video processing. However, the disadvantages of using PSO include a local solution and leading to premature convergence.

2.1 Neighborhood Topologies

PSO topologies outline the neighbor relationship and interaction between particles, which controls the propagation of information in the particle swarm. Because the topologies describe the subset of particles with which each particle exchanges information, the topologies directly affect the swarm's optimization ability and its convergence. The global topology is the most widely used topology where all particles are directly connected to one another. So, each particle can share information with any other particle in the swarm. In the ring topology, the particles are directly connected to their m immediate neighbors. When m = 2, the particle is affected by only its immediately adjacent neighbors. The von Neumann topology operates as a grid structure, where each particle is connected to its four neighbors: top, bottom, left, and right. The random neighborhood topology is such that a random neighborhood of size k is initially created for each particle by choosing k1 other particles randomly without repetition. In subsequent iterations, a particle's neighborhood is recreated in the same way, with a probability of 0.2.

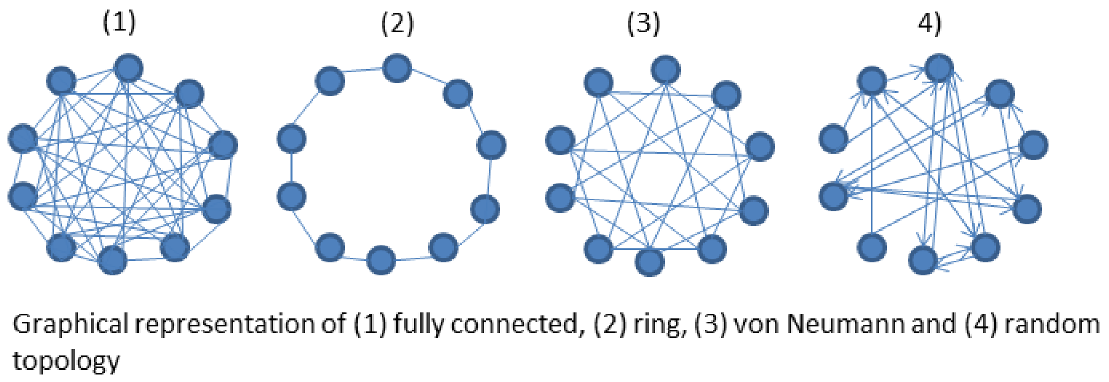

(1)
(2)
(3)
4)
Graphical representation of (1) fully connected, (2) ring, (3) von Neumann and (4) random topology

Figure 2: Different Neighborhood Topologies

2.2 Test Functions

Test functions such as Rosenbrock, Rastrigin, Ackley among others are widely used benchmarks for PSO. These functions evaluate multiple aspects of the algorithm such as the convergence rate, precision, robustness, and general performance.

2.2.1 Rosenbrock Function

The Rosenbrock function is also referred to as the Valley or Banana function. It is shown in the plot above in its two-dimensional form. The function is unimodal, and the global minimum lies in a narrow, parabolic valley. However, even though this valley is easy to find, convergence to the minimum is difficult.

The equation is as follows:

$$f(x) = \sum_{i=1}^{d-1}[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$$

Global Minimum for Rosenbrock:

$$f(x^*) = 0, \text{ at } x^* = (1, ..., 1)$$

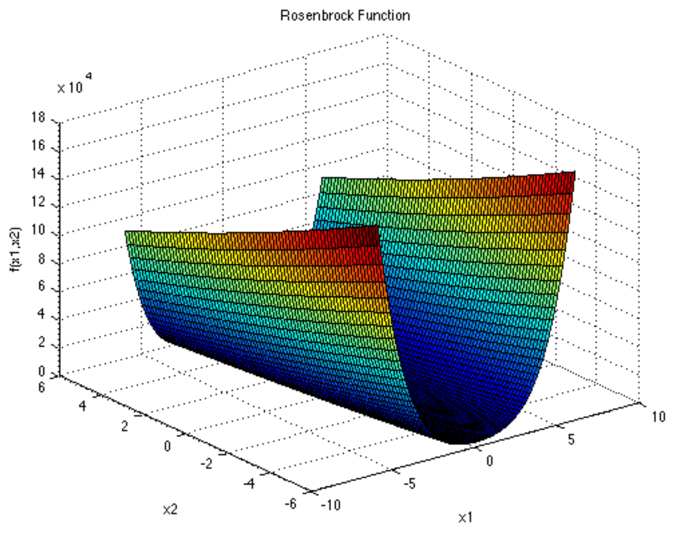


Figure 3: Plot for Rosenbrock varying x1 and x2 values

2.2.2 Rastrigin Function

The Rastrigin function has several local minima. It is highly multimodal, but locations of the minima are regularly distributed. It is shown in the plot below in its two-dimensional form.

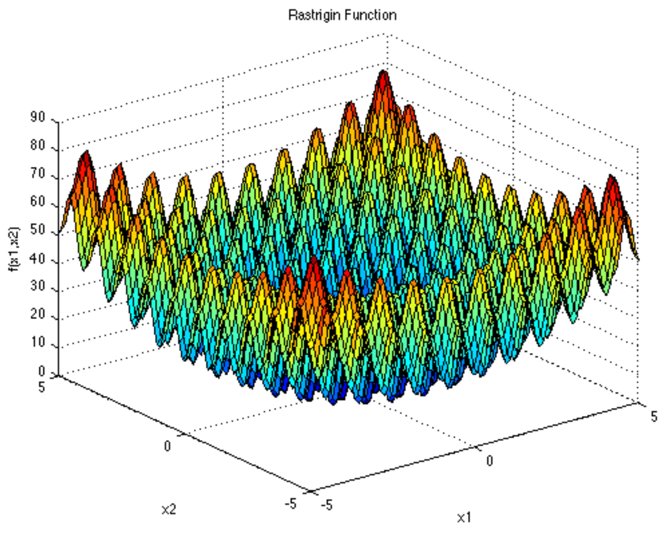


Figure 4: Plot for Rastrigin varying x1 and x2 values

The equation is as follows:

$$f(x) = 10d + \sum_{i=1}^{d} [x_i^2 - 10\cos(2\pi x_i)]$$

Global Minimum for Rastrigin:

$$f(x^*) = 0, \text{ at } x^* = (0, ..., 0)$$

2.2.3 Ackley Function

The Ackley function in its two-dimensional form, as shown in the plot below, is characterized by a nearly flat outer region, and a large hole at the center. The function poses a risk for optimization algorithms, particularly hill climbing algorithms, to be trapped in one of its many local minima.

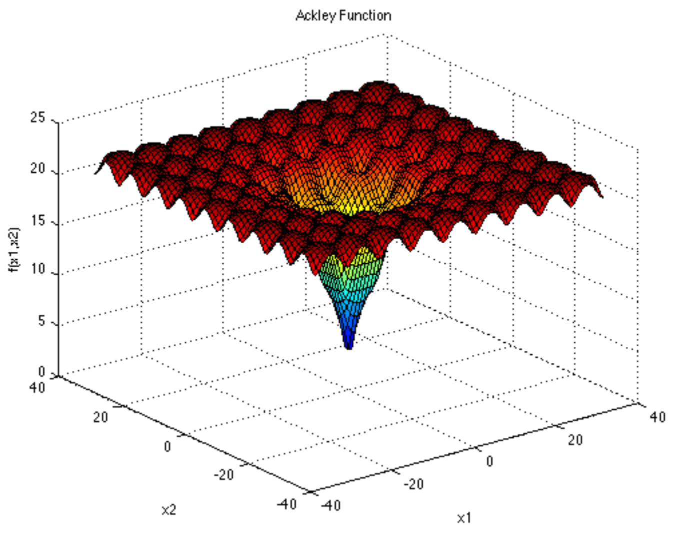


Figure 5: Plot for Ackley varying x1 and x2 values

The equation is as follows:

$$f(x) = -a\exp\left(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}cos(cx_i)\right) + a + \exp(1)$$

Global Minimum for Ackley:

$$f(x^*) = 0,\text{ at } x^* = (0,...,0)$$

3 Experimental Methodology

For our experiments, we manipulated numerous variables - test functions, swarm sizes and more importantly, different neighborhood topologies. For our test functions we used Rosenbrock, Ackley, and Rastrigin. The positions of the particles were initialized between the range of 15.0 and 30.0 for Rosenbrock, between 16.0 and 32.0 for Ackley, and between 2.56 and 5.12 for Rastrigin. Likewise, each particle's initial velocity vector was initialized in a similar manner; the initialization ranges were between -2.0 and 2.0 for Rosenbrock, between -2.0 and 4.0 for Ackley, and between -2.0 and 4.0 for Rastrigin.

For these experiments, we manipulated variables one at a time. Initially we used the test function Ackley, did so with 16 particles, and changed the neighborhood topology. We would get an output of 4 lists (the results obtained from 4 different topologies). The results were printed so that each element in the list of 10 elements corresponded to the best solution found after 1000 iterations. We then changed the number of particles (30), and ran the algorithm for the different topologies with Ackley. Once we recorded results for all swarm sizes (16, 30, 49), and all topologies (global, von Neumann, ring, random), we repeated this methodology for the rest of the test functions. There were 36 possible cases for the different combinations of optimization function, swarm size, and topology. Each of these 36 cases was tested in 30 dimensions and run for 10,000 iterations. This was done 21 times. For each of these cases, we measured the median of the best value obtained by 10,000 iterations over all 21 runs. Additionally, we also recorded the result obtained for every 1000 iterations.

4 Results

	Global			Ring		
# Particles	16	30	49	16	30	49
Ackley	12.37	2.01	0.93	7.26	3.89	1.16
Rosenbrock	6.52	0.29	0.69	13.23	7.13	3.08
Rastrigin	117.40	101.98	93.00	98.00	97.00	76.63

	Von Neumann			Random		
# Particles	16	30	49	16	30	49
Ackley	2.01	1.47E-14	1.47E-14	1.65	1.46E-14	1.47E-14
Rosenbrock	5.45	10.32	9.23	11.94	7.50	10.18
Rastrigin	96.51	93.02	90.54	90.45	84.56	82.63

Table 1: Median best value at iteration 10,000 for all 36 cases

Although later in this section, the graphs will go into more detail on specific results, Table 1.0 generally summarizes the results we obtained by showing the median best value at iteration 10,000 for all 36 cases (3 functions $\times$ 4 neighborhood topologies $\times$ 3 swarm sizes). From this table, we can see that as swarm size increases, the global minimum seems to decrease. This trend generally holds across the four neighborhoods and also across the three functions we tested on. For the Ackley function, we seem to obtain the lowest values for the Random and the Von Neumann topologies. Additionally, for both of these topologies, we get values pretty close to zero with just 16 particles and they continue decreasing over 30 and 49 particles. For the Rosenbrock function, we found that the global topology performed the best, followed by the ring topology. For global, we get a value in the single digits for 16 particles and when we increase the number of particles to 30, the number is pretty close to zero. Similarly, for the ring topology, we get a value in the low teens for 16 particles but by 49 particles, this number has decreased considerably to closer to zero (3.08). Finally, for the Rastrigin function, it was difficult to really identify which topology fared the best as the values obtained were mostly within the same range. Although the results obtained for random were the lowest, there wasn't a notable difference with increasing the swarm size. But for the ring topology, we noticed that there was a significant difference in the results obtained between 16 and 49 particles. So for the Rastrigin function, even though results from the random topology were generally on the lower side, ring topology at 49 particles gave the best results. Overall, we discovered that there are a lot of differences in how different topologies perform. Some topologies find the best value at the lowest swarm size and this number stays mostly constant for bigger swarm sizes, while others don't perform well immediately but end up finding the best value at bigger swarm sizes. Furthermore, this change is dependent on the function used.

4.1 Ackley Results

Figure 6 summarizes the effects of all swarm sizes and all neighborhood topologies using the Ackley Function. Figure 7 breaks down this graph further to show the effects of varying swarm sizes while keeping the topologies constant using the Ackley function. Figure 7a shows effects of varying swarm size on the global topology. It shows that increasing the swarm size produced better values for the swarm minimum. However, the swarm minimum does not change throughout the thousands of iterations. The value converges at the very beginning and stays constant for all iterations. Figure 7c shows effects of varying swarm size on the ring topology.

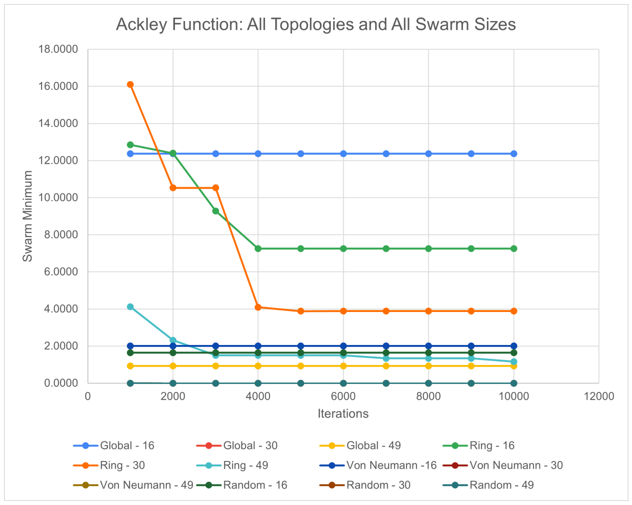
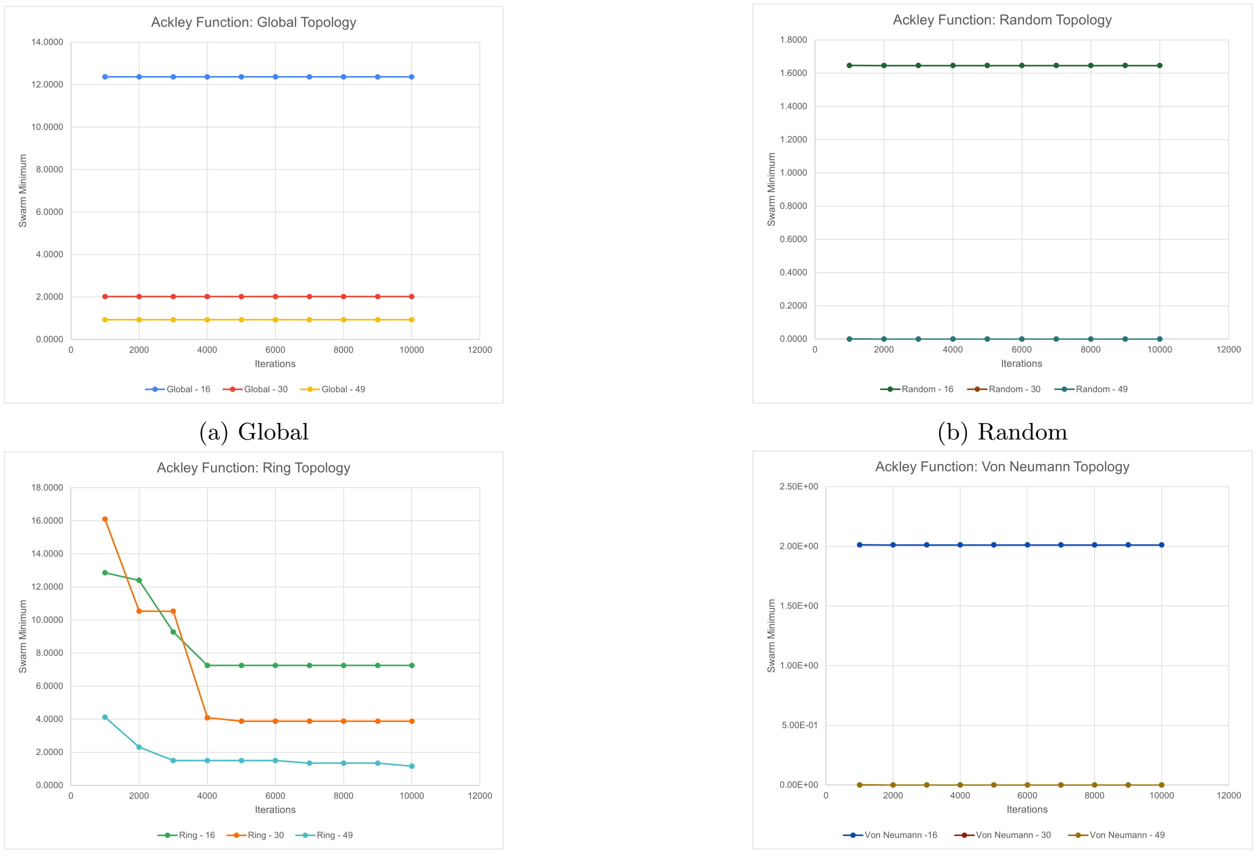


Figure 6: Ackley Master Plot



(a) Global

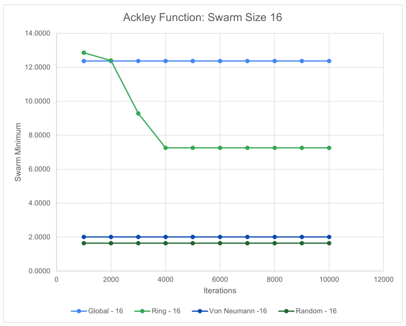
(b) Random

(c) Ring

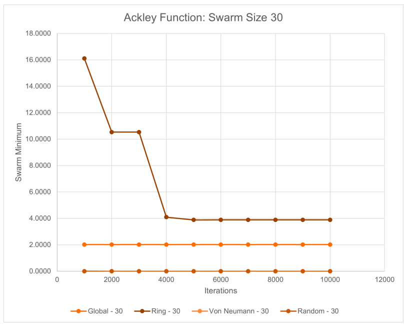
(d) Von Neumann

Figure 7: Different Topologies with Ackley Function

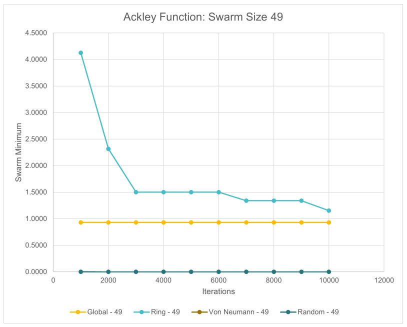
There is a similar trend with better values being produced with the increase in swarm size. However, with the ring topology, the swarm minimum changes over different iterations and this value only converges after four thousand iterations. Figure 7d and 7b show effects of varying swarm size on the von Neumann and random topology respectively. For both the random and von Neumann topologies, we observed similar trends. Comparable to the global topology, for these too, the swarm minimum converges at the beginning and does not change throughout iterations. Additionally, for both of these, there were no differences in



(a) 16



(b) 30



(c) 49

Figure 8: Different Swarm Sizes with Ackley Function

results produced by swarm size 30 and 49.

Now, Figure 8 show the effects of different topologies while keeping the swarm size constant using the Ackley Function. All three figures show that the swarm minimum converges really quickly, within the first thousand iterations, and stays constant for all topologies except the ring topology. For the ring topology, it seems to take upto four thousand iterations for the minimum to converge. This observation remained true for all swarm sizes. Similarly, for all three swarm sizes, von Neumann and Random generated the lowest swarm minimums, and thus the best values. And although the ring topology performs slightly better than the global topology on swarm size of 16, on swarm sizes 30 and 49, there isn't a significant difference between the two.

4.2 Rosenbrock Results

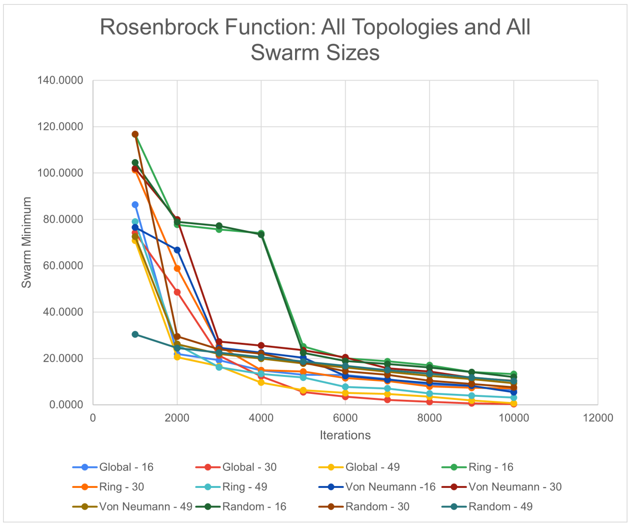
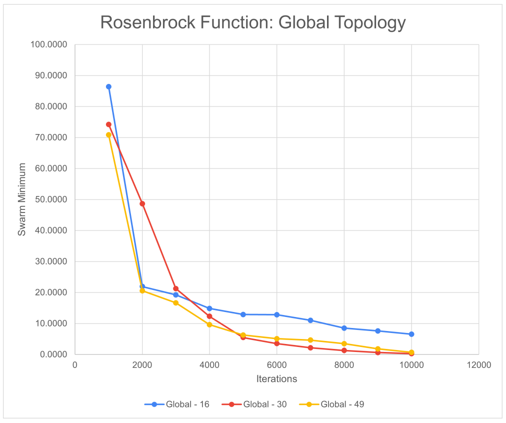
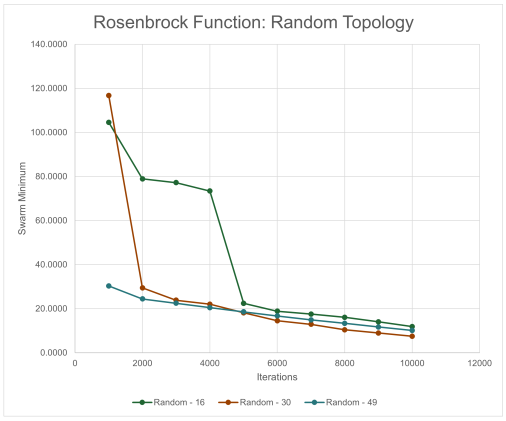


Figure 9: Rosenbrock Master Plot

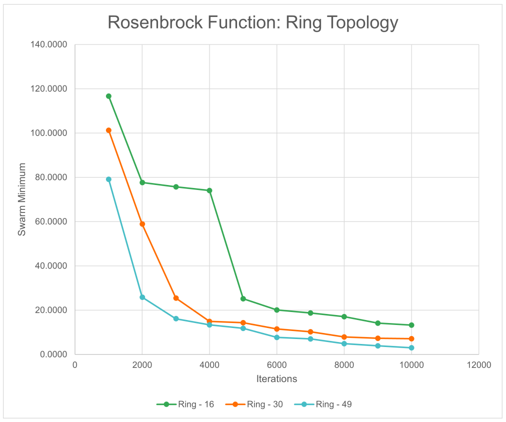
Figure 9 summarizes the effects of all swarm sizes and all neighborhood topologies using the Rosenbrock Function. Figure 10 breaks down this graph further to show the effects of varying swarm sizes while keeping the topologies constant using the Rosenbrock function. Across all four figures, we see the swarm minimum decreasing steadily over the course of the ten thousand iterations. For ring topology and random topology (Figure 10c and 10b, there is a sharp decline in the best value generated between 4000 and 5000 iterations for swarm size 16. In all the figures, swarm size of 30 and 49 perform slightly better than swarm size of 16. Finally, although the performance is generally comparable across all three topologies, global topology performs slightly better by generating values that are almost zero.



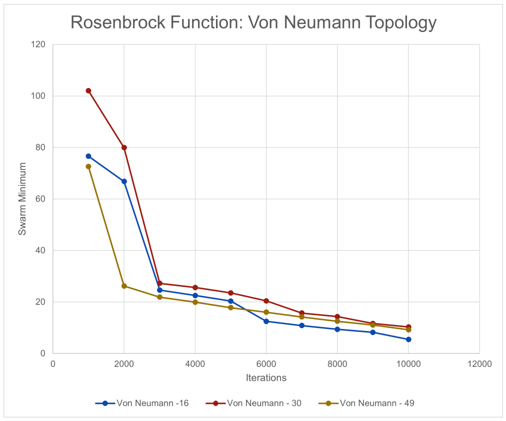
(a) Global



(b) Random

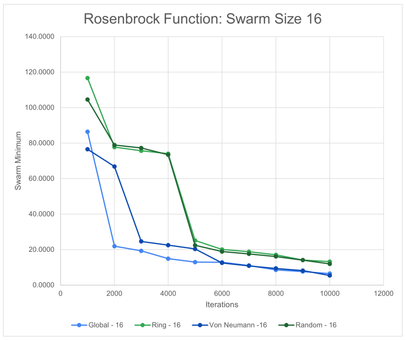


(c) Ring

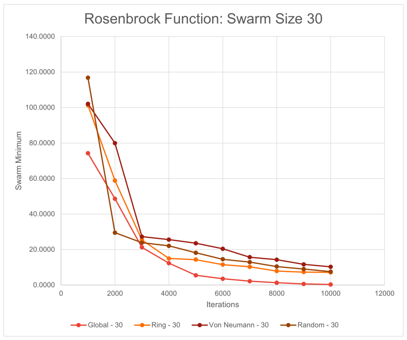


(d) Von Neumann

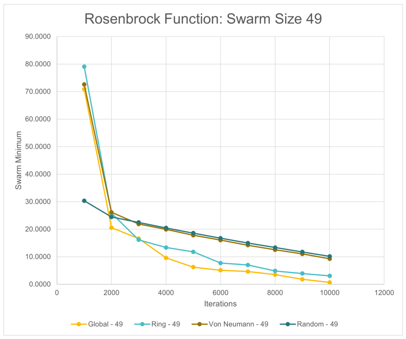
Figure 10: Different Topologies with Rosenbrock Function



(a) 16



(b) 30



(c) 49

Figure 11: Different Swarm Sizes with Rosenbrock Function

Now, Figure 11 shows the effects of different topologies while keeping the swarm size constant using the Rosenbrock Function. These figures suggest that, regardless of the swarm size and the topology, it takes quite some time for the Rosenbrock function to converge, with the swarm minimum continuously declining over iterations. The trends across all three graphs are pretty similar with the global and ring topologies generating the best values for all three swarm sizes.

4.3 Rastrigin Results

Figure 12 summarizes the effects of all swarm sizes and all neighborhood topologies using the Rosenbrock Function. Figure 13 breaks down this graph further to show the effects of varying swarm sizes while keeping the topologies constant using the Rastrigin function. Similar to its performance with Ackley and Rosenbrock, for Rastrigin too, the global topology seems to converge pretty quickly and stays constant for the rest of the iterations. The ring topology performs similarly. Von Neumann and ring topologies show slightly more variation by only converging after 3000 iterations. These observations hold for all swarm sizes. Additionally, von Neumann and ring topologies generate the biggest difference in the best value for each swarm size while for global and ring, these differences are less noticeable. Additionally, as noted with all other cases before, swarm size of 49 produces the best results for all topologies.

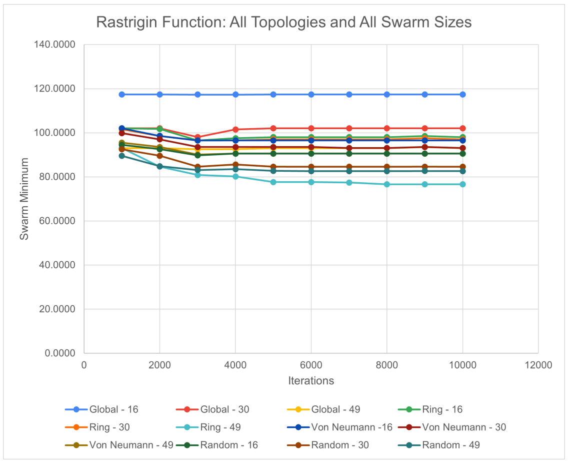
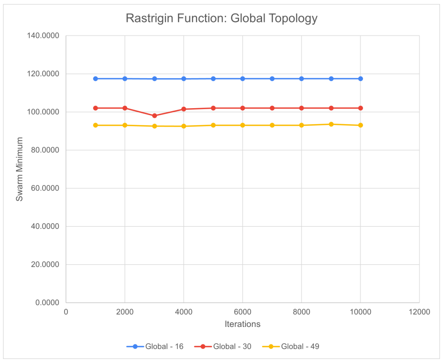
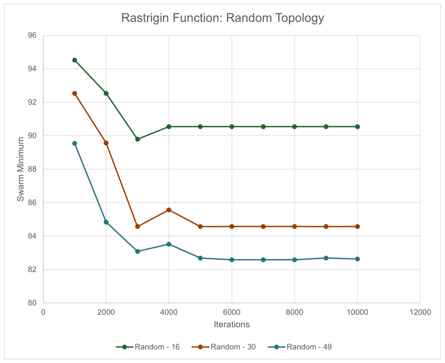


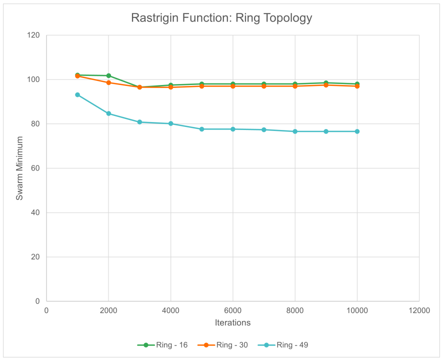
Figure 12: Rastrigin Master Plot



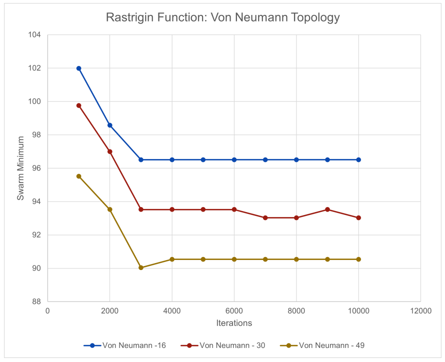
(a) Global



(b) Random



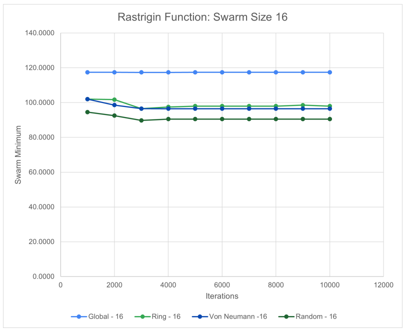
(c) Ring



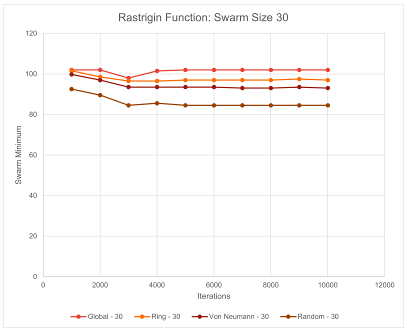
(d) Von Neumann

Figure 13: Different Topologies with Rastrigin Function

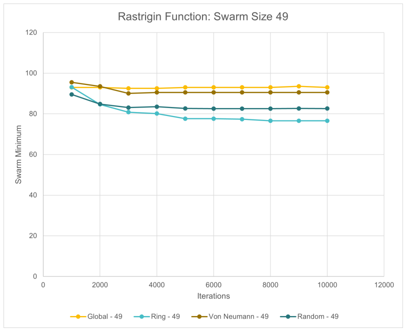
Now, Figure 14 show the effects of different topologies while keeping the swarm size constant using the Rastrigin Function. Figure 14a and 14b show effects of varying topologies while keeping swarm size at 16 particles and 30 particles. In both these cases, the random topology generated the best values while the global topology performed significantly worse. But, for a swarm size of 49 particles, shown in Figure 14c, the ring topology was the best, outperforming even the random topology. Here too, the global topology was the least favorable one.



(a) 16



(b) 30



(c) 49

Figure 14: Different Swarm Sizes with Rastrigin Function

Finally, we observed that compared to the other two functions, all the neighborhood topologies seem to perform significantly worse using the Rastrigin function with regards to the best swarm minimum generated. Both the Ackley and Rosenbrock functions produced better results.

5 Further Work

In terms of minor changes we could have implemented, we found that values generated by Rosenbrock were continually decreasing even at 10,000 iterations. Instead of ending the runs at 10,000 iterations, increasing the number of iterations for Rosenbrock may have yielded better values. Additionally, our implementation of PSO was PSO with a constriction factor. Implementing PSO with inertia weights may have provided further insight. Similarly, we implemented the basic version of the original PSO algorithm but we could have also experimented with other PSO models such as accelerated PSO (APSO) and PSO-Time varying acceleration coefficients (PSO-TVAC). This would allow us to compare and see how the results would have been different.

In terms of the structure of the project, we only used three optimization functions but there are many other available test functions, such as Sphere, Schwefel, Three-Hump Camel. These can also be used for testing the performance of the PSO algorithm. Testing further functions may have further strengthened our analysis. Similarly, we only implemented four neighborhood topologies. There are many others that we did not cover here such as Star, Wheel, Four Clusters, etc. Considering that the whole objective of this project was to assess how different neighborhood topologies perform, maybe our analysis would have benefited from including some of these other topologies too.

Finally, we only implemented the PSO algorithm. But maybe if we had more time, we could have compared PSO with other optimization problems such as Ant Colony Optimization (ACO), Genetic Algorithms (GA), Artificial Bee Colony Optimization (ABC) and others, as well.

6 Conclusion

Our goal was to assess the impact of neighborhood topology on the performance of the Particle Swarm Optimization (PSO) algorithm. After performing 36 tests using different topologies, swarm sizes, and functions, our findings show that with the Ackley function, von Neumann and random topologies performed better compared to the other topologies. For the Rosenbrock function, the global topology performed the best followed by ring topology. Finally, for the Rastrigin function, the ring topology produced the best results. Looking at the final results from our test functions, we can conclude that the topology plays a key role in PSO. Additionally, we also found that certain topology-function combinations perform better than others. Thus, when analyzing an algorithm like PSO, we should take into account both, topologies and the optimization functions used, to assess performance.