Advocating the value of Hybridization Methodologies using Biologically Inspired Algorithms and the Traveling Salesman problem



Elihu Essien-Thompson

A dissertation submitted in partial fulfilment of the requirements of Technological University Dublin for the degree of

M.Sc. in Computer Science (Advanced Software Development)

**April 2022**

I certify that this dissertation which I now submit for examination for the award of MSc in Computing (Advanced Software Development), is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the test of my work.

This dissertation was prepared according to the regulations for postgraduate study of the Technological University Dublin and has not been submitted in whole or part for an award in any other Institute or University.

The work reported on in this dissertation conforms to the principles and requirements of the Institute’s guidelines for ethics in research.

**Signed: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Date: 26/04/2022**

ABSTRACT

AAAA

*(approx. 250-300 words)*

**Key words:** *list 5 to 8 words*

ACKNOWLEDGEMENTS

I would like to express my sincere thanks ……….

*(thank all the people how have assisted you in completing your dissertation. Start with your supervisor, all DIT staff that may have helped, other people can include family and friends, industrial and academic staff from other institution, etc.)*

TABLE OF CONTENTS

[ABSTRACT ii](#_Toc103114132)

[ACKNOWLEDGEMENTS iii](#_Toc103114133)

[TABLE OF CONTENTS iv](#_Toc103114134)

[TABLE OF FIGURES vi](#_Toc103114135)

[TABLE OF TABLES vii](#_Toc103114136)

[1. Introduction 1](#_Toc103114137)

[1.1 Background 1](#_Toc103114138)

[1.2 Research Project 3](#_Toc103114139)

[1.3 Research Objectives 4](#_Toc103114140)

[1.4 Research Methodologies 4](#_Toc103114141)

[1.5 Scope and Limitations 5](#_Toc103114142)

[1.6 Document Outline 5](#_Toc103114143)

[2. LITERATURE REVIEW 6](#_Toc103114144)

[2.1 Combinatorial Optimization and the TSP 6](#_Toc103114145)

[2.2 History of the chosen algorithms 6](#_Toc103114146)

[2.2.1 Genetic algorithm (GA) 6](#_Toc103114147)

[2.2.2 Particle Swarm Optimization (PSO) 7](#_Toc103114148)

[2.2.3 Ant Colony Optimization (ACO) 9](#_Toc103114149)

[2.3 Algorithm Methodology and Variants 11](#_Toc103114150)

[2.4 Statistical Analysis 11](#_Toc103114151)

[3. Design and Methodology 12](#_Toc103114152)

[4. Results, evaluation and discussion 13](#_Toc103114153)

[5. Conclusion 14](#_Toc103114154)

[6. BIBLIOGRAPHY 15](#_Toc103114155)

[Appendix A 20](#_Toc103114156)

TABLE OF FIGURES

No table of figures entries found.

TABLE OF TABLES

# Introduction

## Background

Biologically Inspired Algorithms (BIAs) are a term used to denote a family of algorithms that each arose from an algorithm engineer’s understanding gained through analysis of nature’s solution to common problems. They usually are further subcategorized by their general methodologies like Evolutionary algorithms (using the concept of genetic crossovers) (Back et al., 1997) and Swarm Intelligence (modelled after the behaviours of creatures that operate in swarms like birds, fish and bees; using a team of multiple simplistic agents working together to solve a complex problem) (J. Kennedy & Eberhart, 1995), among many others.

Originally developed by the American engineer John Holland sometime in the 10960s (Coley, 1999; Holland, 1992), one of the earliest occurring members of these Biologically Inspired Algorithms (BIAs) in history is the Genetic Algorithm inspired by Charles Darwin’s theory of evolution through natural selection (Darwin, 1876; Mirjalili, 2019a). Progressing on through the latter quarter of the nineties marked revolutionary findings in the development of more AI technologies like evolutionary computation (Back et al., 1997) and the Artificial Neural Network (Jain et al., 1996) modelled after the inner workings of the brain. These algorithms have found great application in a variety of fields, but few findings made during that time have brought as many revolutionary insights to AI as the emergence of *Swarm Intelligence*.

Swarm intelligence was a method developed to allow exploitation of social behaviours by splitting the computational requirements for performing complex tasks and calculations across a group, or swarm, of simplistic inter-communicating individual agents. Inspiration for the design was taken from the collective behaviour of social organisms such as ants, termites, bees, birds, and fish (Kennedy et al., 2001). Two of the most popular algorithms that arose from implementations of swarm intelligence are the *Ant Colony Optimization* (ACO) and the *Particle Swarm Optimization* (PSO) algorithms (Blum & Li, 2008).

ACO, originally proposed by Dorigo et al. (2006), and PSO, originally proposed by Kennedy et al. (2001), have found great success in application to discrete and continuous domains respectively. ACO has been used as a rough set approach to feature selection (Y. Chen et al., 2010), heart disease prediction and classification (Khourdifi & Bahaj, 2019) and real-time routing problems (Samà et al., 2016). PSO has been used for multi-objective optimization (Delgarm et al., 2016), clustering for high dimensional datasets (Esmin et al., 2015) and scalable optimization through social learning (Cheng & Jin, 2015). Work has been done to bridge the gap in application domains between the two algorithms through some variations in their methodologies (Socha & Dorigo, 2008; Zhong et al., 2007).

Algorithm hybridization is a technique through which the methodologies of the given algorithms are combined in an effort to merge their strengths. Through use and analysis, advantages and drawbacks have been highlighted over the years in many algorithms, like the GA, ACO and PSO (Rohini & Natarajan, 2016; Selvi & Umarani, 2010). As such, nowadays many computer programmers prefer to use the stronger hybrid models, pooling the advantages of their algorithms, for their applications (Luan et al., 2019; Mahi et al., 2015; Mandloi & Bhatia, 2016). Huang et al. (2013) demonstrated some of the techniques through which hybrid models can be built and, through the example of the ACO and PSO hybrid, demonstrated that these hybridization strategies each came with a different level of efficacy. Unfortunately, even though hybridization for these BIAs has popularity in literature, not much attention has been drawn to analysing their hybridization strategies.

In a lot of the studies mentioned above, comparative analysis for the algorithms has simply been completed by first choosing an application domain that the algorithms would be run on, and then comparing the results drawn from the study. An example would be combinatorial optimization problems, for which, the ACO algorithm has established great accomplishments (Huang et al., 2013; Papadimitriou & Steiglitz, 1998). With regards to combinatorial optimization problems, the Traveling Salesman Problem (TSP) is one of the most important and widely applicable issues that has arisen (Yousefikhoshbakht, 2021). That is because it sets the foundation for building efficient routing systems; systems that we are all constantly in connection with today. The TSP has been one of the many tools used over the years to benchmark the performance of and demonstrate the merits that the use of BIAs can bring (S.-M. Chen & Chien, 2011).

## Research Project

The majority of the literature read in this study that dealt with hybrid models has only ever considered and documented a single hybrid construction methodology. The study done by Huang et al. (2013) was the only one found that did otherwise. Perhaps through further research into this field, patterns and possibly heuristics can be gleaned to direct the choice of hybrid methodologies justified by highlighted characteristics found in the base algorithms used. Extracting these patterns could, like the revolutionary *Swarm Intelligence*, open up new avenues for our understanding of AI.

Unfortunately, the requirements for such a study far outreach the scope of what this dissertation can accomplish. Instead, this project aims to advocate the value of hybridization and hybridization strategies. The rationale is that since there exists variation in the components that can make up a given base algorithm, then logically, there should also exist variation in the hybrids that can be produced from that algorithm.

The research question that this project aims to answer is:

“When a statistical analysis is done on the results between hybrid and base models, which of the chosen Biologically Inspired Algorithms is found to be the fastest to converge on the most optimal solution to the Traveling Salesman Problem, given standardized population size, number of maximum iterations, and a statistical significance threshold of 0.05?”

*The chosen BIAs for this study are the GA, PSO and ACO algorithms, along with the 3 hybrid models created from mixing them [ACO/GA, PSO/GA and PSO/ACO].*

## Research Objectives

This project comes with two main objectives. The first objective is to demonstrate some of the variations in design found for the chosen BIAs. The second, and more important objective, is to establish the benefits of hybridization by showing that a hybrid algorithm performs better than the base algorithm. To achieve this joint objective, the following goals were defined:

1. To research the chosen BIAs in order to find and understand some of the most popular variations that have occurred in their design.
2. To define appropriate parameters and statistical tests to be used in this study to justify any conclusions drawn from the final comparative analysis.
3. To perform the comparative analysis of the base algorithms run against their hybrid versions to extract the conclusion(s) of the study.

## Research Methodologies

To fulfil the research goals from Section 1.3, two research methods are utilized: secondary research (through a literature review) and empirical research (through implementation and evaluation of the findings from the review). The breakdown of the approach taken to solve those research goals mentioned in Section 1.3 is as follows:

1. Perform a literature review to research the chosen BIAs in order to find and understand some of the most popular variations that have occurred in their design
2. Perform a literature review on the most commonly used statistical tests to understand and justify any statistical tests performed in the study
3. For each of the main variants extracted from step 1, conduct empirical research on the efficacy of that variant within the chosen problem domain (TSP) by implementing them in Python and running them against the TSP maps generated to find the best representative for each of the algorithms that would be used in the final experiment.
4. Create the 3 Hybrid algorithms using the optimum methodologies extracted from step 3.
5. Using the test(s) chosen from step 2 and representatives chosen from steps 3 and 4, conduct the final comparative analysis of the algorithms documenting any conclusion(s) drawn.

## Scope and Limitations

This study touches on interesting topics in the theory of computation like discrete and single-objective optimization, graphic algorithm analysis, and the theory of randomized search heuristics. It also discusses machine learning theories, like artificial intelligence, biologically-inspired optimization, multi-agent reinforcement learning and evolutionary algorithms. Finally, mathematical topics are also touched on, like mathematical modelling and optimization.

Unfortunately, due to monetary limitations over quarantine, it was decided to carry out the study using a borrowed college laptop having an Intel® Core™ i5-10210U CPU @1.60GHz 2.11GHz processor, a 16BG ram capacity, and a 64-bit Operating System. Due to the number of variants needing to be tested, the experiments completed took a lot of time to run and, given also the learning curve of understanding the algorithms and programming languages used, the experiments often had to be repeated after any algorithm discoveries were made that required a code change. It was decided very early on, for efficiency’s sake, that the earlier experiments would be completed using simpler maps (TSP maps of 10 cities to visit), and only the final experiment would be run on the more complex maps (TSP maps of 50 cities).

Also, as mentioned earlier in Section 1.2, the time and recourses allocatable for this project were a limitation that spurred the decision not to tackle directly the original gap highlighted found in the body of knowledge (*optimum hybridization strategies*).

## Document Outline

* Chapter 2:
* Chapter 3:
* Chapter 4:
* Chapter 5:
* Chapter 6:

# LITERATURE REVIEW

## History of the chosen algorithms

### Genetic algorithm (GA)

Charles Darwin’s Theory of Evolution through natural selection in his “On the Origin of Species” book (1876), though not completely factual as noted by the evolutionary biologist Stuart Newman (Mazur, 2008) and Gordons’s (1999) findings on “The Concept of Monophyly”, has inspired many through analysis of its applications. The theory is based on the observation: ‘*survival of the fittest*’, in which, fitter and more capable individuals of a population naturally achieve higher survival rates in their given environments, providing them longer lifespans and more opportunities to pass on their superior genetic codes to the next generation. The weaker members of the population would typically achieve lower chances to pass on their inferior genetics, eventually being completely overwritten from the genetic history by fitter candidates over the progressing generations. Darwin’s theory hinged on the concept of variation; that there is a range of differences between the genetic makeup of the individuals in the population, which when accumulated through the principles above would be able to push organisms past the barrier of species toward something completely different, perhaps new, but ultimately better.

Genetic algorithms are a family of computational models that draw inspiration from Darwin’s evolutionary theory known to have been originally introduced and investigated by the American engineer John Holland sometime in the 1960s (Coley, 1999; Holland, 1992). Mirjalili (2019) reports that Holland’s genetic algorithm was “one of the first population-based stochastic algorithms proposed in history”. Using a chromosome-like data structure and recombination operators to simulate the mechanics of DNA reproduction, these algorithms have been applied to a very broad range of problems often offering optimized potential solutions.

The algorithm offers a wide application domain because as long as the given problem can be encoded as a chromosome-based population and a function for evaluation of individual fitness (or attractiveness), the GA can be utilized. The ease in flexibility plays is a part of the reason why the GA remains one of the most popular evolutionary algorithms in literature (Mirjalili, 2019a) with various applications found like the automatic design of convolutional neural networks for image classification (Sun et al., 2020), as a solver for systems of second-order boundary value problems (Arqub & Abo-Hammour, 2014), optimization of cogeneration plant systems (Ahmadi & Dincer, 2010) and optimizing back-propagation (BP) neural networks (Ding et al., 2011), among many others.

### Particle Swarm Optimization (PSO)

Arguably since the invention of the electronic computer (possibly even earlier), scientists and philosophers alike pondered over the similarities between computer programs and minds. Similar to minds, computers demonstrated the capability to process symbolic information, derive conclusions from premises, and recall stored information when appropriate. They reasoned that the capability of minds to host intelligence gave direction to the possibilities for computers, hence, birthing the great quest for Artificial Intelligence (AI) (Kennedy et al., 2001).

Progressing from the latter quarter of the nineties marked revolutionary findings in the development of AI technologies like the GA, evolutionary computation (Back et al., 1997), and the Artificial Neural Network (ANN) (Jain et al., 1996). However, the social psychologist James Kennedy and his associates (Kennedy et al., 2001) observed a stereotype creeping into the general understanding of AI at the time that was limiting the understanding of AI at the time. They noted that early AI researchers understood the measure of intelligence as the ability to solve large, complex, and sometimes multipart, problems quickly. Due to the variety in methods to approach many problems, building an intelligent computer program that finds the best choice required and motivated them to think of a number of clever methods. They developed ‘logical shortcuts’, called *heuristics*, that speed up the process in a manner that was applicably reusable. The programs developed by the researchers were simply outstanding at problem-solving, calculation and memory retention but were found to fail at simpler things like conversation and face recognition. This was due to the continuously growing number of variables still needing to be addressed in the problem domains it was being applied to. There was always something else that could go wrong.

Kennedy (2001) notes that those “early AI researchers had made an important assumption, so fundamental that it was never stated explicitly nor consciously acknowledged”. AI at the time was modelled on the vision of a single disconnected person capable of coolheadedly handling the situations posed to them using the information and logical reasoning stored in their brain. However, they argued that if human intelligence was the intended model, then this model of understanding was devoid of an important comportment/behaviour involved in human reasoning and development: *Socialization*.

In real social interaction, not only information but also rules, tips, and also methods of processing information are exchanged. They observed further social behaviours which were “the norm throughout the animal kingdom” in biological examples like Fish schooling, birds flocking and bugs swarming. These behaviours occurred not only for copulation purposes but included important necessities for the population like “having a thousand eyes” to keep watch for predators and searching for food, among other advantages.

Their book “Swarm Intelligence” cited earlier, introduced the concept of exploiting social behaviours by splitting the computational requirements of a system across a group or ‘swarm’ of intercommunicating individuals. Gaining inspiration from the natural examples mentioned earlier, they proposed a model called *Particle Swarm Optimization* which differed from the popular evolutionary computation methods at the time because its population members, named *particles*, were first initialized with stochastically assigned positions and velocities, and then flown through the problem space in search of a solution. The stochastic mechanisms implemented in the algorithm gave it a “lifelike” appearance (J. Kennedy & Eberhart, 1995) as the particles buzzed around the search space resembling a swarm of mosquitoes. They believed that this observed behaviour, along with the description of each particle being, in essence, a mass-less and volume-less mathematical abstraction that can be called a point when stationary, deemed the terms *particle* and *Particle Swarm Optimization* fitting descriptors for their model.

Due to its simplistic and effective design, PSO has gained a lot of popularity over the years (Blum & Li, 2008; Eberhart & Shi, 2001; J. Kennedy & Eberhart, 1995) and has found applications in many domains such as scalable optimization for social learning (Cheng & Jin, 2015), clustering for high dimensional data sets (Esmin et al., 2015), and multi-objective optimization (Delgarm et al., 2016). With regards to the TSP, studies were undertaken to adapt the algorithm to these discrete domains (Zhong et al., 2007). One interesting algorithm adaptation was the one proposed by Wang et al. (2003) and further improved by Yousefikhoshbakht (2021), where a series of swap sequences were used to represent vectors. This adaptation was chosen as the PSO focus for this study.

### Ant Colony Optimization (ACO)

Around the same time as the PSO, the ACO algorithm was also developed. It gained inspiration from the study done by Pierre-Paul Grassé, the French entomologist in the 40s and 50s of the 20th century, who shone a light on some interesting findings observed in some species of termites. He observed the reactions of these termites to something he called “significant stimuli” and found that those reactions themselves could also operate as significant stimuli for other insects in the colony, including the insect that produced them. This special type of communication found in these species was termed *stigmergy* and it was described with two main characteristics (Dorigo et al., 2006; Salman et al., 2020):

* It is an indirect, non-symbolic form of communication using the environment as a medium (i.e., communication through modification of the environment).
* The stigmergic information created is local (i.e., can only be accessed when in the vicinity/locus in which it was released).

Diagram

Description automatically generatedSince then, stigmergy has been observed in many other species including ant colonies. In ant species, as the members travel in search of, or returning from a food source, they deposit a chemical along the trails they traverse called *pheromone*. Other ants, upon inspecting a trail, can perceive these pheromones and, as a response, tend to follow the trail containing higher pheromone levels. As they traverse their chosen path, they also add their own deposited pheromone trail to the path, further increasing its pheromone concentration and the ‘attractiveness’ of this trail to successive ants on arrival. The remarkable efficacy of this exploratory pattern was demonstrated by the thorough investigation performed by Deneubourg et al. (1990). In their, soon to be well known, “double bridge experiment”, they introduced a diamond-shaped bridge between the ant nest and a chemically unmarked arena for the ants to explore. This provided the ants with a binary left/right choice in such a way that the “dynamics of their cumulative choice [could] be easily quantified”. They noted that the ant’s stigmergic system exploited the positive feedback loop such that it, “after initial fluctuation, rapidly leads to one of the two forks becoming more or less completely preferred to the other” and eventually the whole colony converges on the use of only one of the bridges.

Figure 1: Double-Bridge Experiment

Diagram

Description automatically generatedGoss et al.(1989) expanded on this study by adding a food source to the arena and differing the size of the two bridges in the experiment. Again, the axis of entry is the same 30° on both sides of the bridges (total 60° between bridges) to minimize ‘loop back’ ant journeys and so that the forager has no preference for one or branch or the other based on position. In their experiment, at first, the ants were choosing equally between the short and long but “abruptly, some minutes later, one branch becomes visibly preferred”. The ants, at first choosing stochastically, the shortest bridge were the first to reach the nest, so on return the probability that they take again take the shorter path is higher as there is no pheromone trail attracting them to the longer path yet until those on that path finally arrive. Their choice then reinforces that pheromone trail as they deposit more on the way back, positively affecting bias towards this trail for all successive ants (Blum & Li, 2008). Their proposed model for that observed behaviour became the main source of inspiration for developing the Ant Colony Optimization algorithm we know today (Dorigo et al., 2006).

Figure 2: Different length bridges

Over time, ACO has become one of the most popular biologically inspired algorithms in literature (Blum & Li, 2008) and has been used to solve many graph-based or graph adapted combinatorial optimization problems. It has found applications in areas like feature selection using a rough set approach (Y. Chen et al., 2010), heart disease prediction and classification (Khourdifi & Bahaj, 2019), scheduling problems (Deng et al., 2019), and real-time routing problems (Samà et al., 2016). In fact, work has also been done to increase the applications of the algorithm to include problems based on continuous domains (Socha & Dorigo, 2008).

## Combinatorial Optimization Problems and the TSP

When the goal is the optimization of problems occurring with qualitative, or discrete, variables (e.g., attributes, states, or values), the solution to the problem consists of arranging those components in such a way that it minimizes, or maximizes, the desired result. In some cases, that goal includes eliminating some of those components as well, meaning that the number of elements to rearrange also becomes part of the problem. This process of seeking the best possible solution within a finite set of possibilities is what is called combinatorial optimization and a problem solved through the arranging of its propositions is a combinatorial optimization problem (Kennedy et al., 2001). Combinatorial Optimization problems all come with a goal that is optimized towards and an objective function through which the solutions proposed can be critiqued. With the example of a company, having a machine that drills holes into printed circuit boards, that wants the machine to complete its job as fast as possible by minimizing the time taken to move the drill from one point to another, the problem can be explained as “what is the most efficient route for the machine to take?”, and the objective function would correspondingly be a measure of the distance travelled for any route/solution proposed. That is because, in this example, the total distance travelled serves as the metric through which a given solution can be critiqued against the optimization goal (Korte & Vygen, 2012).

Some examples of combinatorial optimization problems are Bin-Packing (Delorme et al., 2016), Job-Shop Scheduling (Zhang et al., 2011) and Boolean Satisfiability (Soeken et al., 2010). However, one of the most well-known Combinatorial Optimization Problems is the Traveling Salesman Problem (TSP) (Yousefikhoshbakht, 2021). The challenge of the TSP can be defined by the question: “Given a map of cities to visit and the distances between each pair of cities, what is the shortest round trip that can be made from a given origin city, visiting each city on the map exactly once, and returning back to your starting position?”. The problem is characterised by two main conditions:

1. Each city must be visited exactly one time
2. The trip must conclude with a loop back to the starting position

With this in mind, the optimum route/solution to the TSP can be described as ordering an itinerary of cities to visit in such a way that the sum of distances traversed while following the itinerary returns the smallest possible value.

Equation 1: TSP Distance calculation

Where means the distance between cities and , if the location of city and , then . Since the goal is to find the smallest possible total distance, the calculation in **Equation 1** can be inverted so that the answer returned can be used as a score for the proposed solution.

Equation 2: TSP Objective Function

According to **Equation 2**, the smaller the total distance travelled in a given TSP solution, the larger the score that would be awarded to that solution. Both **Equations 1** and **2** are usable for the objective function and are both compatible with the algorithms used. However, **Equation 2** was chosen for this study because I believe it best captures the meaning of the TSP with regard to the members of the algorithm’s population.

## Optimization Methodology

When dealing with optimization problems, the array of possible valid solutions is often illustrated as a *search space,* or *search landscape,* which exists on an -dimensional plane, for which, each point on that search space represents a possible valid solution and the dimensions of the plane correspond to the different variables existing in that problem domain (Mirjalili, 2019b). Solutions existing in relatively close locations to one other in the search space would receive similar scores from the objective function because of the close proximity of their input variable values which denoted their dimensional location. **Figure 3** demonstrates an example search space showing a plane of possible solutions using a combination of input variables *x* and *y* represented as the *length* and *width* dimensions, and the score given from the objective function for those possible inputs as the *height* dimension.

Chart

Description automatically generated

Figure 3: Example Search Space/Landscape (Mirjalili, 2019b)

In Section 2.2, a mathematical model demonstrating how a TSP solution can be tested was constructed. The objective function in **Equation 2**, takes in a possible solution (sequence of cities) as an input and returns a score, through which, the efficacy of that solution can be measured. Hence, the role that optimization algorithms play with respect to this objective function is to devise an input solution to be supplied to the function, that returns the highest score possible. Given the conditions included in the TSP problem definition and the nature of combinatorial optimization, the number of possible valid solutions that can be accepted into the objective function is finite. So in other words, the role of the optimization algorithms is to traverse the finite search space seeking the highest peak (a location/position for which the objective function returns the highest score) (Blum & Li, 2008).

The algorithms operate by locating and exploring promising regions within the search space. But, when a peak is found, how is it determined whether this location is the highest in the entire search space? This important consideration of *Local Optimum* vs *Global Optimum* is critical to the optimization algorithm development process as it determines the adequacy of a given algorithm design.

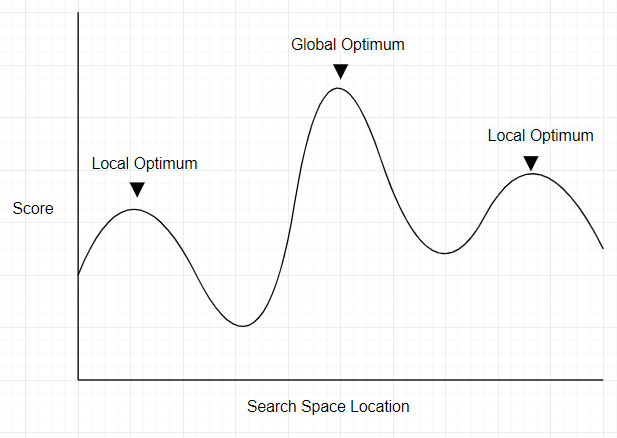


Figure 4: Local vs Global Optimums

The algorithms focused on in this study, provide different solutions to this problem. In GAs, *Selection Pressure* (SP) refers to the “degree to which the better individuals are favoured: the higher the selection pressure, the more the better individuals are favoured” (Miller & Goldberg, 1995). The SP is the driving force for improvement over succeeding generations in the GA and it is a primary influence when it comes to GA convergence. If the algorithm lends too much focus on the best individuals of a particular generation, (the SP is too high, i.e., is too focussed on a certain region of the search space), disregarding the potential gain from considering the other members, then the algorithm converges toward those member’s solutions ardently as the optimum. This ‘tunnel-vision’ risks convergence on a local rather than global optimum. So, care needs to be taken when analysing the SP of a given GA design.

The Swarm Intelligence algorithms PSO and ACO on the other hand, operate by balancing between two mechanisms, *exploration* (diversification) and *exploitation* (intensification) (Mirjalili, 2019b; Thangaraj et al., 2011). As suggested by the name, Exploration involves diversifying the regions searched by the members of the swarm by making frequent or large changes to the solutions that they compose. This throws a stochastic element into the algorithm that pulls the members away from the currently targeted ‘best location’, allowing them the opportunity to find and explore other potentially better avenues of the search space. On the other hand, Exploitation is the mechanism that offers the opposite behaviour, through which, all members of the algorithm converge towards the optimum solution.

For all algorithms traversing the search space in seeking an optimum, their search is brought to a conclusion when certain criteria are reached and the best solution is given. The two most popular criteria for search termination are convergence, when the majority of the members of the algorithm’s population converge on a single solution (Miller & Goldberg, 1995), and through use of a search counter, when the maximum number of algorithm iterations allowed is reached (Ahmadi & Dincer, 2010). It should be noted that this ‘best solution’ does not always mean the global optimum, but rather the best optimum found when the stop criteria is reached.

## Algorithm Structure and Variants

### Genetic Algorithm (GA) overview

GA begins with an encoding of the problem domain as a list of chromosomes representing an initial population. These chromosomes are an arbitrary set of trial solutions often randomized to provide unique starting points for each member of the population within the search space. Mirjalili (2019a) notes that techniques like Gaussian random distribution can be used to maximize diversity in the initial population while others like Johnson & Rahmat-Samii (1997) do not find that extra step necessary given the robustness offered by the complete algorithm.

After initialization, a combination of two techniques called *evaluation* and *fitness allocation* is used to award each member a measure of ‘attractiveness’ (also called fitness) in such a way that those chromosomes which represent a better solution to the target problem are given more chances to 'reproduce' than those chromosomes which are poorer solutions. The Evaluation Function provides a means to measure the performance of a given individual regarding a set of parameters extracted from the problem domain (i.e., the TSP objective function given in Equation 2) which is then taken by the Fitness Function and transformed it into an allocation of reproductive opportunities. The ‘attractiveness’ of any given individual is typically assigned relative to the current population (Whitley, 1994). Using this combined process, Selection Operator chooses the best individuals from the population and compiles them into a mating pool. It is the Breeding Operator’s task then to combine the genetic components of those chromosome members in that mating pool in order to make the next generation.

At this point, the earlier mentioned issue of Selection Pressure (SP) comes into play. Emphasis must not be overly placed on these best individuals when allocating mating opportunities until it is more certain that their chromosome patterns are optimum. This is especially apparent after the initialization step because of the reasonably low chances of finding the Global Optimum through random initialization. Rather, the selection and breeding operators aim to progressively extract and combine favourable parts of the genetic codes of the population while discarding the unfavourable. As the generations go by, through this iterative process, these favourable chromosome components would gradually become more prominent in the population set until a consensus is eventually made on an optimum component set.

As part of the last step of the Breeding Operation, before the creation of the next generation is finalized, is the introduction of a very important component of the GA: *mutation*. So far, the GA process begins with a varied initial population and, through its selection and breeding mechanisms, isolates desirable gene sequences within the chromosomes to focus on, making these components gradually more prominent as generations progress. However, it should be noted that there is no guarantee of having the globally best genetic components within the initial populations of the algorithm, restricting the algorithm’s search space; hinting toward the importance of mutation. Mutation can be seen as the operator charged with maintaining the genetic diversity of the population as it aims to preserve the diversity embodied in the initial generation. It does this by introducing new information into the genetic sequence, allowing the population to ‘leapfrog’ over potential sticking points (Coley, 1999).

In the final step of the GA, mutations are randomly assigned under an appropriately low percentage to allow more variability in the search space. As the algorithm iterates through the generations, members of the population increase in fitness until a concluding best solution is found after the stop criteria is/are reached (Whitley, 1994).

Diagram

Description automatically generated

Figure 5: GA Flowchart

### The GA Selection Operator: Fitness Function Variants

The main variation in GA composition techniques occurs within the Selection Operator. As mentioned earlier, this operator performs its selection using a combination of two techniques, one of which is the fitness function. Earlier it was also highlighted that the role of this function is convert the score given from population evaluations into an allocation of reproductive opportunities. In other words, to convert the evaluation score into measure of *fitness* (or attractiveness) as a new *fitness score*. This can be done in a number of ways:

#### Roulette Wheel Selection (RWS)

As the title suggests, the concept of natural selection is simulated using a roulette wheel type selection process. The *fitness score* in RWS refers to the number of slots allotted on the wheel to each member of the population, and it is calculated relative to each member’s evaluation score. The probability of selecting a member of the population in RWS can be viewed as the probability of the selection pointer landing on that member after a roulette wheel, with the size of the segments for each member proportional to their fitness score, is spun (Razali & Geraghty, 2011). With the list of fitness values for all members of the population the selection probability for any individual is:

Equation 3: RWS Selection Probability

To calculate that fitness score, often times the evaluation scores of all members of the population are first normalized for algorithm consistency, and then scaled according to how large the wheel is desired to be. For example, if 10 slots are the largest size segment that can be allotted on the wheel and 1 is the smallest, then those normalized values ranging between 0 and 1 can be scaled up to the range 1-10 using **Equation 5**.

Equation 4: Normalization Formula

*Where is the score, min(x) is the smallest score in the group, and max(x) is the largest score in the group.*

Equation 5: Converting ranges

*Where x is the value to convert, ‘new’ is the new range, 'cur’ is the current range that x is on, min( ) returns is the lower boundary value of the given range, and length( ) returns the length of the given range.*

Whitley (1994) offered the suggestion to deal with any remainder values generated after using **Equations 4** and **5**, which was used in this study. He suggested to use those remainders as a probability for offering a bonus slot to that member. In that case, the final equation needed for calculating the fitness score would be:

Equation 6: Fitness Score Remainders

*Where is the fitness score, r is its decimal values, floor( ) returns the value given rounded down, and prob( ) returns 1 with a probability of the value given or else it returns 0.*

#### Stochastic Universal Sampling (SUS)

#### Rank Based Sampling

#### Roulette Wheel Sampling

## Statistical Analysis

# Design and Methodology

# Results, evaluation and discussion

# Conclusion

# BIBLIOGRAPHY

Ahmadi, P., & Dincer, I. (2010). Exergoenvironmental analysis and optimization of a cogeneration plant system using Multimodal Genetic Algorithm (MGA). *Energy*, *35*(12), 5161–5172. https://doi.org/10.1016/j.energy.2010.07.050

Arqub, O. A., & Abo-Hammour, Z. (2014). Numerical solution of systems of second-order boundary value problems using continuous genetic algorithm. *Information Sciences*, *279*, 396–415. https://doi.org/10.1016/j.ins.2014.03.128

Back, T., Hammel, U., & Schwefel, H.-P. (1997). Evolutionary computation: Comments on the history and current state. *IEEE Transactions on Evolutionary Computation*, *1*(1), 3–17. https://doi.org/10.1109/4235.585888

Blum, C., & Li, X. (2008). Swarm Intelligence in Optimization. In C. Blum & D. Merkle (Eds.), *Swarm Intelligence: Introduction and Applications* (pp. 43–85). Springer. https://doi.org/10.1007/978-3-540-74089-6\_2

Chen, S.-M., & Chien, C.-Y. (2011). Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques. *Expert Systems with Applications*, *38*(12), 14439–14450. https://doi.org/10.1016/j.eswa.2011.04.163

Chen, Y., Miao, D., & Wang, R. (2010). A rough set approach to feature selection based on ant colony optimization. *Pattern Recognition Letters*, *31*(3), 226–233. https://doi.org/10.1016/j.patrec.2009.10.013

Cheng, R., & Jin, Y. (2015). A social learning particle swarm optimization algorithm for scalable optimization. *Information Sciences*, *291*, 43–60. https://doi.org/10.1016/j.ins.2014.08.039

Coley, D. A. (1999). *An Introduction To Genetic Algorithms For Scientists And Engineers*. World Scientific Publishing Company.

Darwin, C. (1876). *The Origin of Species: By Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life* (6th ed.). Cambridge University Press. https://doi.org/10.1017/CBO9780511694295

Delgarm, N., Sajadi, B., Kowsary, F., & Delgarm, S. (2016). Multi-objective optimization of the building energy performance: A simulation-based approach by means of particle swarm optimization (PSO). *Applied Energy*, *170*, 293–303. https://doi.org/10.1016/j.apenergy.2016.02.141

Delorme, M., Iori, M., & Martello, S. (2016). Bin packing and cutting stock problems: Mathematical models and exact algorithms. *European Journal of Operational Research*, *255*(1), 1–20. https://doi.org/10.1016/j.ejor.2016.04.030

Deneubourg, J.-L., Aron, S., Goss, S., & Pasteels, J. M. (1990). The self-organizing exploratory pattern of the argentine ant. *Journal of Insect Behavior*, *3*(2), 159–168. https://doi.org/10.1007/BF01417909

Deng, W., Xu, J., & Zhao, H. (2019). An Improved Ant Colony Optimization Algorithm Based on Hybrid Strategies for Scheduling Problem. *IEEE Access*, *7*, 20281–20292. https://doi.org/10.1109/ACCESS.2019.2897580

Ding, S., Su, C., & Yu, J. (2011). An optimizing BP neural network algorithm based on genetic algorithm. *Artificial Intelligence Review*, *36*(2), 153–162. https://doi.org/10.1007/s10462-011-9208-z

Dorigo, M., Birattari, M., & Stutzle, T. (2006). Ant colony optimization. *IEEE Computational Intelligence Magazine*, *1*(4), 28–39. https://doi.org/10.1109/MCI.2006.329691

Eberhart, & Shi, Y. (2001). Particle swarm optimization: Developments, applications and resources. *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546)*, *1*, 81–86 vol. 1. https://doi.org/10.1109/CEC.2001.934374

Esmin, A. A. A., Coelho, R. A., & Matwin, S. (2015). A review on particle swarm optimization algorithm and its variants to clustering high-dimensional data. *Artificial Intelligence Review*, *44*(1), 23–45. https://doi.org/10.1007/s10462-013-9400-4

Gordon, M. S. (1999). The Concept of Monophyly: A Speculative Essay. *Biology and Philosophy*, *14*(3), 331–348. https://doi.org/10.1023/A:1006535524246

Goss, S., Aron, S., Deneubourg, J. L., & Pasteels, J. M. (1989). Self-organized shortcuts in the Argentine ant. *Naturwissenschaften*, *76*(12), 579–581. https://doi.org/10.1007/BF00462870

Holland, J. H. (1992). *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press.

Huang, C.-L., Huang, W.-C., Chang, H.-Y., Yeh, Y.-C., & Tsai, C.-Y. (2013). Hybridization strategies for continuous ant colony optimization and particle swarm optimization applied to data clustering. *Applied Soft Computing*, *13*(9), 3864–3872. https://doi.org/10.1016/j.asoc.2013.05.003

Jain, A. K., Mao, J., & Mohiuddin, K. M. (1996). Artificial neural networks: A tutorial. *Computer*, *29*(3), 31–44. https://doi.org/10.1109/2.485891

Johnson, J. M., & Rahmat-Samii, V. (1997). Genetic algorithms in engineering electromagnetics. *IEEE Antennas and Propagation Magazine*, *39*(4), 7–21. https://doi.org/10.1109/74.632992

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN’95 - International Conference on Neural Networks*, *4*, 1942–1948 vol.4. https://doi.org/10.1109/ICNN.1995.488968

Kennedy, J. F., Eberhart, R. C., & Shi, Y. (2001). *Swarm intelligence*. Morgan Kaufmann Publishers.

Khourdifi, Y., & Bahaj, M. (2019). Heart Disease Prediction and Classification Using Machine Learning Algorithms Optimized by Particle Swarm Optimization and Ant Colony Optimization. *International Journal of Intelligent Engineering and Systems*, *12*. https://doi.org/10.22266/ijies2019.0228.24

Korte, B., & Vygen, J. (2012). *Combinatorial Optimization: Theory and Algorithms* (Vol. 21). Springer Science & Business Media. https://doi.org/10.1007/978-3-642-24488-9

Luan, J., Yao, Z., Zhao, F., & Song, X. (2019). A novel method to solve supplier selection problem: Hybrid algorithm of genetic algorithm and ant colony optimization. *Mathematics and Computers in Simulation*, *156*, 294–309. https://doi.org/10.1016/j.matcom.2018.08.011

Mahi, M., Baykan, Ö. K., & Kodaz, H. (2015). A new hybrid method based on Particle Swarm Optimization, Ant Colony Optimization and 3-Opt algorithms for Traveling Salesman Problem. *Applied Soft Computing*, *30*, 484–490. https://doi.org/10.1016/j.asoc.2015.01.068

Mandloi, M., & Bhatia, V. (2016). A low-complexity hybrid algorithm based on particle swarm and ant colony optimization for large-MIMO detection. *Expert Systems with Applications*, *50*, 66–74. https://doi.org/10.1016/j.eswa.2015.12.008

Mazur, S. (2008, October 11). *The Origin of Form Was Abrupt Not Gradual—Archaeology Magazine Archive* [Interview Article]. Archaeology Archive - A Publication of the Archaeological Institute of America. https://archive.archaeology.org/online/interviews/newman.html

Miller, B. L., & Goldberg, D. (1995). Genetic Algorithms, Tournament Selection, and the Effects of Noise. *Complex Syst.*, *9*(3), 193–212.

Mirjalili, S. (2019a). Genetic Algorithm. In S. Mirjalili (Ed.), *Evolutionary Algorithms and Neural Networks: Theory and Applications* (pp. 43–55). Springer International Publishing. https://doi.org/10.1007/978-3-319-93025-1\_4

Mirjalili, S. (2019b). Introduction to Evolutionary Single-Objective Optimisation. In S. Mirjalili (Ed.), *Evolutionary Algorithms and Neural Networks: Theory and Applications* (pp. 3–14). Springer International Publishing. https://doi.org/10.1007/978-3-319-93025-1\_1

Papadimitriou, C. H., & Steiglitz, K. (1998). *Combinatorial Optimization: Algorithms and Complexity*. Courier Corporation.

Razali, N. M., & Geraghty, J. (2011). *Genetic Algorithm Performance with Different Selection Strategies in Solving TSP*. 6.

Rohini, V., & Natarajan, A. M. (2016). Comparison of Genetic Algorithm with Particle Swarm Optimisation, Ant Colony Optimisation and Tabu Search based on University Course Scheduling System. *Indian Journal of Science and Technology*, *9*(21). https://doi.org/10.17485/ijst/2016/v9i21/85379

Salman, M., Garzón Ramos, D., Hasselmann, K., & Birattari, M. (2020). Phormica: Photochromic Pheromone Release and Detection System for Stigmergic Coordination in Robot Swarms. *Frontiers in Robotics and AI*, *7*. https://www.frontiersin.org/article/10.3389/frobt.2020.591402

Samà, M., Pellegrini, P., D’Ariano, A., Rodriguez, J., & Pacciarelli, D. (2016). Ant colony optimization for the real-time train routing selection problem. *Transportation Research Part B: Methodological*, *85*, 89–108. https://doi.org/10.1016/j.trb.2016.01.005

Socha, K., & Dorigo, M. (2008). Ant colony optimization for continuous domains. *European Journal of Operational Research*, *185*(3), 1155–1173. https://doi.org/10.1016/j.ejor.2006.06.046

Soeken, M., Wille, R., Kuhlmann, M., Gogolla, M., & Drechsler, R. (2010). Verifying UML/OCL models using Boolean satisfiability. *2010 Design, Automation Test in Europe Conference Exhibition (DATE 2010)*, 1341–1344. https://doi.org/10.1109/DATE.2010.5457017

Sun, Y., Xue, B., Zhang, M., Yen, G. G., & Lv, J. (2020). Automatically Designing CNN Architectures Using the Genetic Algorithm for Image Classification. *IEEE Transactions on Cybernetics*, *50*(9), 3840–3854. https://doi.org/10.1109/TCYB.2020.2983860

Thangaraj, R., Pant, M., Abraham, A., & Bouvry, P. (2011). Particle swarm optimization: Hybridization perspectives and experimental illustrations. *Applied Mathematics and Computation*, *217*(12), 5208–5226. https://doi.org/10.1016/j.amc.2010.12.053

V.Selvi & Dr.R.Umarani. (2010). Comparative Analysis of Ant Colony and Particle Swarm Optimization Techniques. *International Journal of Computer Applications*, *5*. https://doi.org/10.5120/908-1286

Wang, K.-P., Huang, L., Zhou, C.-G., & Pang, W. (2003). Particle swarm optimization for traveling salesman problem. *Proceedings of the 2003 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.03EX693)*, *3*, 1583-1585 Vol.3. https://doi.org/10.1109/ICMLC.2003.1259748

Whitley, D. (1994). A genetic algorithm tutorial. *Statistics and Computing*, *4*(2), 65–85. https://doi.org/10.1007/BF00175354

Yousefikhoshbakht, M. (2021). Solving the Traveling Salesman Problem: A Modified Metaheuristic Algorithm. *Complexity*, *2021*, e6668345. https://doi.org/10.1155/2021/6668345

Zhang, G., Gao, L., & Shi, Y. (2011). An effective genetic algorithm for the flexible job-shop scheduling problem. *Expert Systems with Applications*, *38*(4), 3563–3573. https://doi.org/10.1016/j.eswa.2010.08.145

Zhong, W., Zhang, J., & Chen, W. (2007). A novel discrete particle swarm optimization to solve traveling salesman problem. *2007 IEEE Congress on Evolutionary Computation*, 3283–3287. https://doi.org/10.1109/CEC.2007.4424894

# Appendix A