To: Dr. Christine Cranford

From: Student A

Subject: Project 4 Reflective Statement

Date: April 29, 2019

In Project 5, the main problem I am trying to address is Travelling Salesman Problem, which is a famous optimization problem in mathematics and computer science. The intended audiences are CEO and Organizational Board Members. The purpose of this project is comparing two different algorithms and showing how they are complementary. The desired outcome is to show the possibility to divide TSP based on its features into small parts and use appropriate algorithm to solve each part.

I changed the message between proposal and presentation because the audience and the media have changed. The audiences were math and computer science students who have some knowledge about algorithms. The audiences now don’t have so much knowledge so I will include fever details of the algorithm. The media was proposal which could include more details to explain the idea in depth. It is presentation so the message now is easier to understand and less technical. The main purpose also changes to show the audience a general idea instead of a plan.

By explaining two algorithms, I will ensure the audience understands the strength and the limitation of each algorithm. Then in the presentation, I will clarify the message in the last few slides by comparing the advantages and disadvantages of those two. On the other hand, in the proposal, the message is briefly mentioned in the Introduction and Recommendation. The evidences are shown in the Solutions part.

In term of visualization, the important texts are bold to distinguish from the rest. In the GA’s crossover part, different background colors are used to differentiate two parents. I also included some images to show the process, especially when the process is hard or requires long sentences to explain. Tables are used to compare the two algorithms so that the audience can see the differences without the effort to memorize.

One risk is the same as the one in the proposal, which is only explaining two algorithms and ignoring other algorithms whose performance may be better. An additional risk is deciding what information to include in the presentation based on the assumption about the audience.

To: Dr. Christine Cranford, CEO

From: Wenting Zheng

Subject: Nature-Inspired Computing to Travelling Salesman Problem

Date: April 8, 2019

**Introduction**

Traveling Salesman Problem is a well-known problem in mathematics and computer science because of its importance in planning and logistics. The most intuitive application of Traveling Salesman Problem is planning the airport tours and power cables. However, it can also be used in genetic engineering and electrical engineering, such as DNA sequencing. Due to its popularity and large applications, Clay Mathematics Institute is providing a million prize to the solution of the TSP. This paper will explore how two nature-inspired algorithms perform in the Traveling Salesman Problem and compare their advantages and disadvantages.

**Problem**

Traveling Salesman Problem(TSP) is a combinatorial optimization problem. Given a set of cities and the distance between each pair, our goal is to find the shortest route that visits all the cities without repetition and returns to the original city. The distance between cities does not necessarily mean how far the two cities are physically apart. It represents the cost to travel from one city to another, so the cost from city A to city B sometimes doesn’t equal to the cost from city B to city A. For example, the flight cost between two cities usually are not the same and the time needed is not equivalent. Therefore, not all the graphic representation of TSP is consistent with the Triangle Inequality Theorem, which states the sum of the length of any two sides is greater than the third one in a triangle. In Figure 1, the left graph is based on Euclidean Distance, so the cost between the two cities are fixed. The right graph is based on the traveling cost, so the cost between two cities vary. In this paper, we use the Euclidean Distance one for simplicity.



**Background**

The Traveling Salesman Problem was first defined in the 1800s by W.R. Hamilton and Thomas Kirkman and haven’t been solved now. The problem doesn’t seem to be difficult if we just use a set of four cities, but it will be exponentially expensive to compute all the solutions of a larger amount of cities. Given *n* cities and their location, the salesman starts with one of them. Then, he has *(n - 1)* cities left to choose. After he moves to the second city, *(n - 2)* cities are left. This process goes on until he finishes the tour, so we have possible tours. Because the tours are not distinguished by the direction, we also need to divide the number by two. Therefore, given *n* cities, there will be possible solutions.



Wheat and Chessboard Problem is a famous mathematical problem. In the story, the king promised to give the man any reward he wants if he wins the chess game. The man simply asked the put one grain on the first square of the chessboard, two on the second, and four on the third. The amount is doubled every time and he wants to have the sum of the grains of wheat. The sum is grains of wheat. The weight is about 1,199,000,000,000 metric tons. which is about 1,645 times the global production of wheat in 2014. This demonstrates the rapid growth of the exponential function . The factorial function , on the other hand, even grows faster than exponential function , so it’s nearly impossible to compute all the solutions and compare them to find the best one as the number of the cities increases.

There is a way to reduce the time complexity to , which is much less than factorial function , but the time needed is still exponential and expensive. Due to the cost to compute all the solutions, it’s effective to find a relatively good solution with some algorithms.

**Objective**

The purpose of the proposal is to explore some well-known nature-inspired algorithms to the Traveling Salesman Problem. These algorithms are inspired by the natural behaviors, such as animal migration and gravity, and many of them “have been attracting considerable attention for their good performance” (Fang, Li, Zhang, Hu, 2015). In this paper, we will specifically focus on two nature-inspired algorithms, Ant Colony Optimization(ACO) and Genetic Algorithm(GA). By comparing the advantages and disadvantages, we will identify some situations where using one algorithm is more efficient than the other. The goal is not finding the optimal solution to TSP, but to probe the differences between the algorithms so that we use less time to find a relatively better solution.

**Solutions**

Ant Colony Optimization(ACO) obviously mimics ants’ behavior of finding the shorter route from the nest and food source with pheromone-based communication. The leftmost image of Figure 3 is the simplest situation where there is no obstacle in the way from nest to food. The ants just move towards one direction. The second image is when the pheromone starts to work. If a stone is put on the way, the ants don’t know which way is shorter. They will each randomly chose a path and deposit some pheromone along the way. The pheromone is a chemical substance that evaporates as time passes, so the ants will get some information about the tour based the remnant pheromone. Interestingly, ants are more likely to choose the path with more pheromone, which indicates a shorter length, but it’s not a guaranteed choice. This randomness ensures the high probability of choosing the current best tour as well as the opportunity of exploring the new tours.



Genetic Algorithm(GA) simulates the process of natural selection with operations such as selection, crossover, and mutation. Every generation consists of a certain number of individuals. Selection operation retains the individuals with better performance in one generation. The crossover, also called recombination, is used to combine two individuals’ genetic information and generate a completely new offspring.



Mutation operation alters one or more genetic values to maintain diversity in the process.



The searching process is parallel in ACO. Ants can search without interrupting each other but also can get feedback from the previously deposited pheromone. However, ACO is more complex and time-consuming than GA. ACO has far more parameters than GA and each can affect the performance of the algorithm, so it’s relatively hard to explore its effect and find the best combination of the parameters. Even you find a good combination, it’s more likely this combination only works in certain situations. Time for the converging process, where all ants agree on one route, is another disadvantage of ACO. On the other hand, GA is easier, cheaper and faster to implement because it has fewer parameters. Nevertheless, the result is very dependent on the initial population. For example, if the initial parents perform well, it’s more likely for their children to have better performance than those whose parents’ performance is not good. Another limitation of GA is its performance in larger TSP size. ACO usually has better results than GA when the number of cities increases.

**Scheduling**

| Time | Event |
| --- | --- |
| January 2020 | Implement ACO in Python and test it with randomly generated TSP |
| February 2020 | Implement GA in Python and test it with randomly generated TSP |
| March 2020 | Use the TSP with the existing solution to compare the generated solution to the best solution. This process will refine programs |
| April 2020 | Explore how the size of the cities impact the performance of each algorithm |

**Budget**

| Description | Cost |
| --- | --- |
| Employing a student to help | $3,000 |
| Programming tools | $300 |
| Total | $3,300 |

**Recommendations**

There are many researchers trying ways to find a solution, but no one has succeeded yet. ACO and GA are mainly used for solving other combinatorial optimization problems, so they have the potential to solve the TSP. Some research found ACO performs better in large-size TSP than GA and GA is easier to implement and faster to generate the result. It’s possible that some features of the TSP, such as the problem size, will impact on the solution and the choice of the algorithm. Our project will focus on exploring how final performance is influenced by the problem features and algorithm parameters. Therefore, we can divide the large-scale TSP into smaller ones and use different algorithms to save time and also find a better solution.

**Reference**

Wei Fang, Xiaodong Li, Mengjie Zhang, and Mengqi Hu, “Nature-Inspired Algorithms for Real-World Optimization Problems,” *Journal of Applied Mathematics, vol*. 2015, Article ID 359203, 2 pages, 2015. <https://doi.org/10.1155/2015/359203>.

Varshika Dwivedi, Taruna Chauhan, Sanu Saxena, Princie Agrawal, “Traveling Salesman Problem using Genetic Algorithm,” *International Journal of Computer Applications.* 2016, <https://pdfs.semanticscholar.org/c31b/fd16935da83e419d631245272d7838262308.pdf>

To: Dr. Christine Cranford

From: Student A

Subject: Project 4 Reflective Statement

Date: April 8, 2019

In Project 4, the problem I am trying to explore is the Traveling Salesman Problem, which is a well-known problem in mathematics and computer science. The intended audiences are mainly mathematics and computer science students and anyone who is interested in nature-inspired algorithms or Traveling Salesman Problem. The main purpose is to explore another possibility in TSP by using different algorithms to solve a part of TSP based on their performance. The desired outcome is to find a range in TSP where one algorithm generally performs better than the other.

The importance of TSP is clearly indicated by the 1 million rewards. To be more specific, the TSP is associated with the time and money in many different fields, such as optimization and planning. The assumption I made about the TSP is its complexity and difficulty because it hasn’t been solved for a long time. It doesn’t really change as I work through the proposal. Using an appropriate algorithm usually takes less computation time to find a well-performed result. Obviously, Ant Colony Optimization(ACO) and Genetic Algorithm(GA) are not the only algorithms in TSP.

There are researchers who use hybrid algorithms in TSP, which has a similar idea. Instead of using one algorithm to solve TSP, we are trying to divide the large TSP into different parts based on the features. Then we will use the approporiate algorithm to solve the small parts. This proposal mainly uses ACO and GA to compare. By researching the two algorithms, we can conclude they have very different advantages and disadvantages, so their applicable ranges may vary depending on the features of TSP. For example, if GA usually performs better than ACO when city size is less than 50, we can choose GA given a problem size of 20. Problem size is the simplest feature of TSP and there are many other features that may influence how these two algorithms perform in TSP.

The other alternative solutions are Miller-Tucker-Zemlin and Particle Swarm Optimization. However, they are either hard to visualize or don’t have too much in common. ACO and GA are both nature-inspired algorithms which have good performance in other similar optimization problems. TSP is identified as an optimization problem, so it’s possible for these two algorithms to find the optimal solution to TSP.

I decided the information and how to present it based on the variance of the audience. I assume the audience has some knowledge about the TSP and some algorithms. However, not all of them are experts so I decided to include some background of TSP and description of algorithms. I also choose to include some images to visualize the process and difference, such as the growth of exponential and factorial functions.

The main risk is narrowing the tremendous amount of algorithms down to two nature-inspired algorithms. This can be limited and is very likely to neglect other better algorithms.