CHAPTER IV
ANALYSIS AND DESIGN

4.1. Analysis

4.1.1. Use Case Diagram

For providing the workflow, use case diagram will be provided. That explains more about how the programs worked. Also requirements for the programs to worked, user will need to fill the parameters. In order for the system to provide the results of the program.

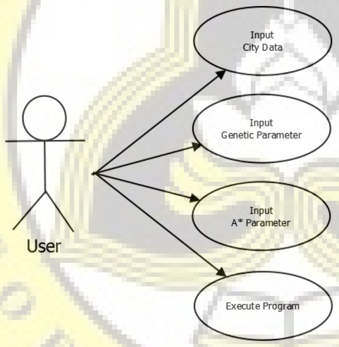

Figure 1. User Use Case Diagram

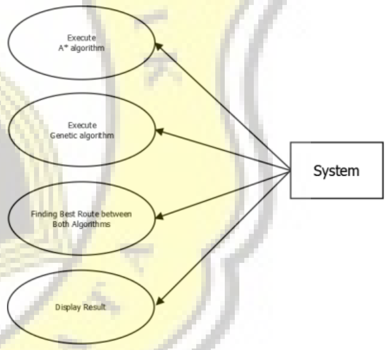

Figure 2. System Use Case Diagram

Users will input the cities that needs to be explored. Users will also fill the parameters required for both algorithms (mutation rate, elitism, heuristic value finder type, and many others). After required parameter has been filled, users will execute the program to start calculating. And system will execute and display final results of both algorithms.

4.2. Design

4.2.1. A* Flowchart

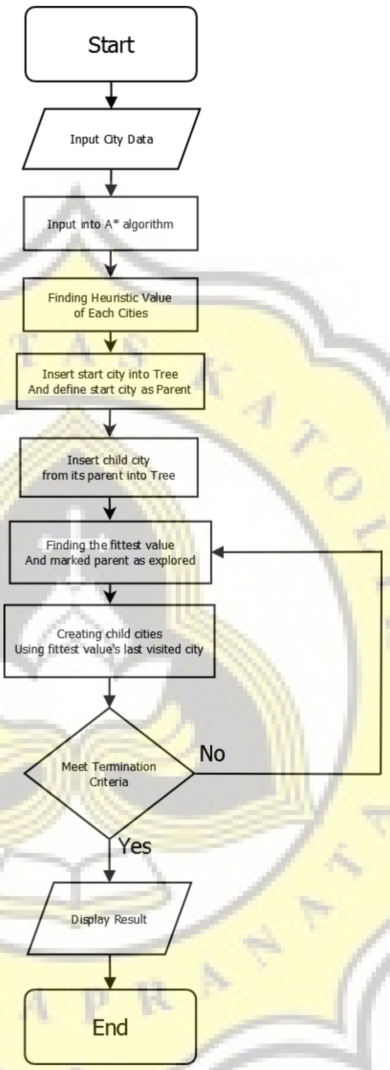
Start
Input City Data
Input into A* algorithm
Finding Heuristic Value of Each Cities
Insert start city into Tree And define start city as Parent
Insert child city from its parent into Tree
Finding the fittest value And marked parent as explored
Creating child cities Using fittest value's last visited city
Meet Termination Criteria
No
Yes
Display Result
End

Figure 3. Flowchart of A* algorithm

After city data has been inputted, then system start finding the heuristic of each city value. Then, insert the start city and its children into the tree while marked the explored city as visited. Fittest value will be found and makes it create another child city using fittest value's last visited city. When criteria do not meets, it loops back to finding value again. And if criteria meets then results displayed.

4.2.2. Genetic Flowchart

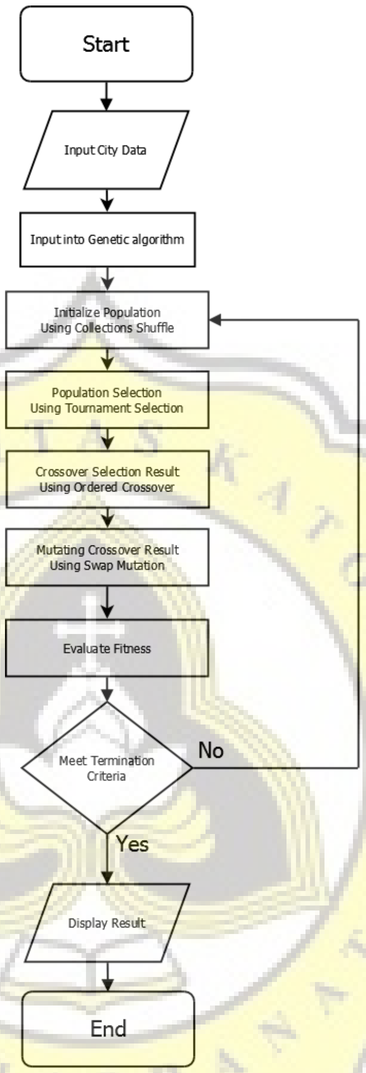
Start
Input City Data
Input into Genetic algorithm
Initialize Population Using Collections Shuffle
Population Selection Using Tournament Selection
Crossover Selection Result Using Ordered Crossover
Mutating Crossover Result Using Swap Mutation
Evaluate Fitness
Meet Termination Criteria
No
Yes
Display Result
End

Figure 4. Flowchart of Genetic algorithm

After input cities data, then initialize population using Collections Shuffle. Then choose the best selection using Tournament Selection. Used the selection and ordered crossovers them with another selection. Mutates the crossover result with Swap Mutation to find optimal solution. Doing the loop of initialization, selection, crossover, and mutation until generations reach its limit. And displays the best result of finding the value using Genetic algorithm.

4.2.3. A* Class Diagram

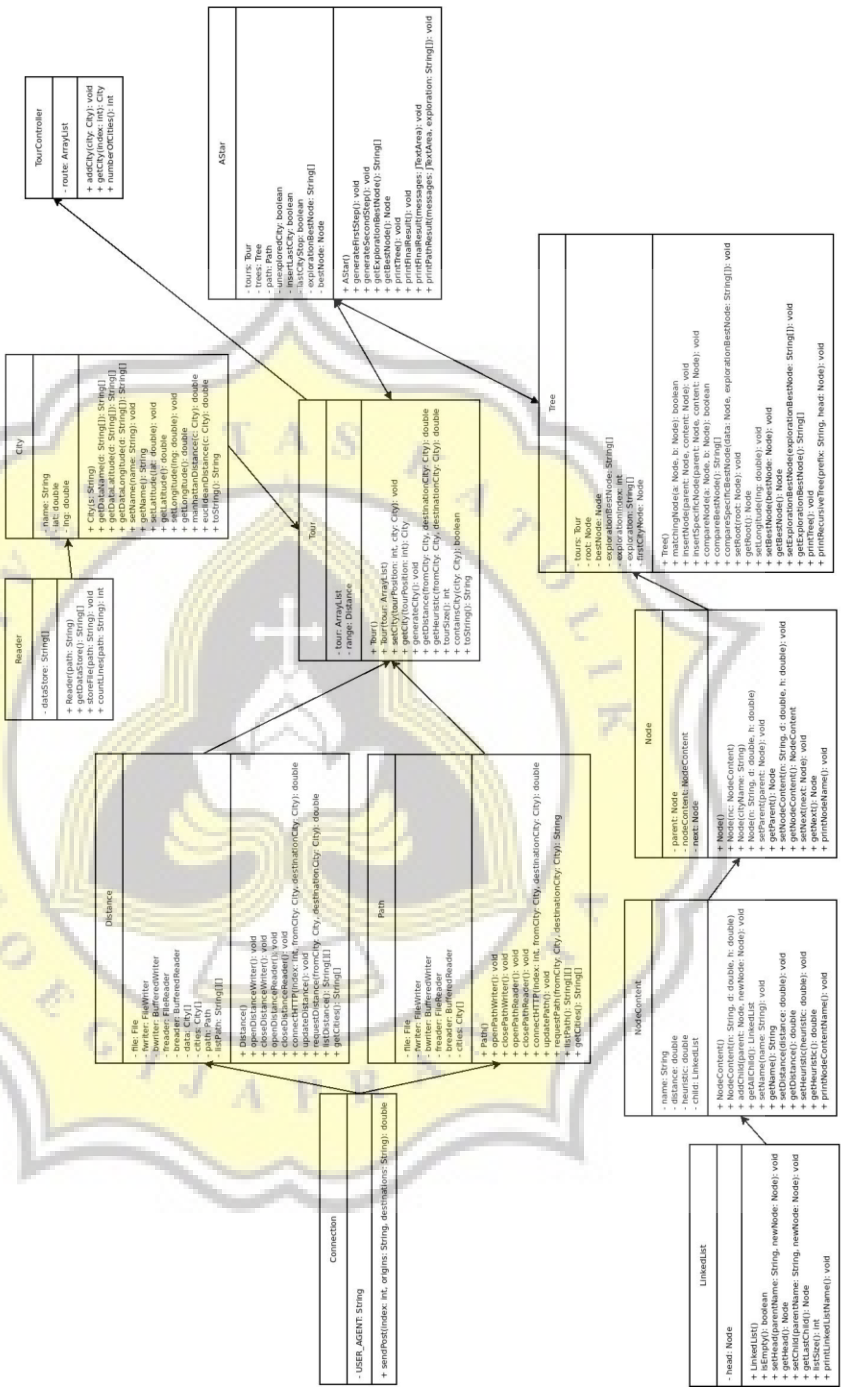

Figure 5. Class Diagram of A* algorithm

This class diagram explains that class Connection that works to sending HTTP request are used by class Distance and Path. This class Distance is class that contains real distance of city. And class Path is class that contains pathway that needs to be taken when visiting city.

In other condition, Class City needs class Reader in order to read database text file, to provide information of the city. Which class City function will provide the stored data when programs needs to get information of the City. These class City, class Distance, class Path will be processed by class Tour. Function of the class Tour is to calculate distance and heuristic, also gives the value to another class when needed, and handle back-end service for the user. Between them placed TourController to handle front-end service of the user.

For the data structure, class Node contains NodeContent and self-references of Node class in order to move into next node. Contents of NodeContent class have information of city name, stored distance value, stored heuristic value, and LinkedList class works as class child that can directing to multiple Node. Then for the data structures, there are class Tree that will contains multiple Node class. Also class Tree holds the formula to inserts and searches best candidates.

This class Tree and class Tour will be processed with class AStar, where the main process and rules of A* algorithm will be applied into the process. And also become bridge between both classes to be accessed.

4.2.4. Genetic Class Diagram

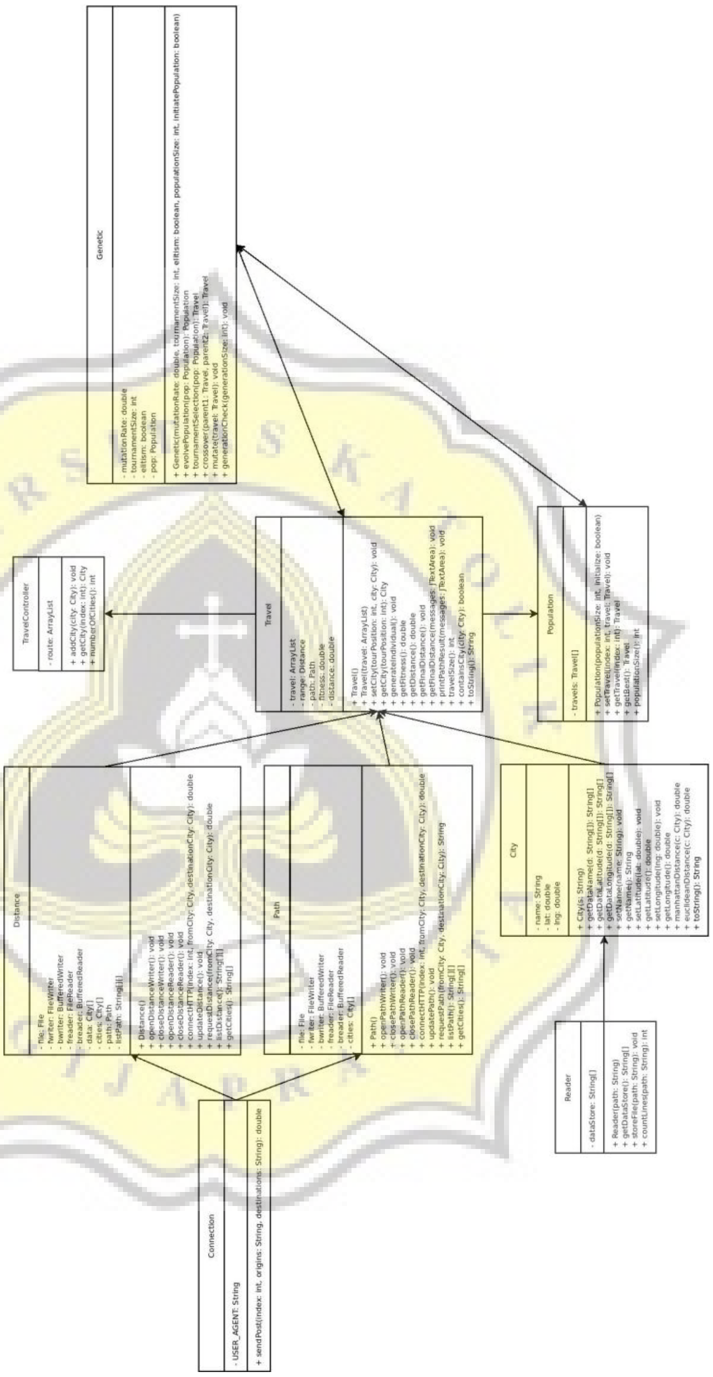

Figure 6. Class Diagram of Genetic algorithm

This class diagram explains that class Connection that works to sending HTTP request are used by class Distance and Path. This class Distance is class that contains real distance of city. And class Path is class that contains pathway that needs to be taken when visiting city.

In other condition, Class City needs class Reader in order to read database text file, to provide information of the city. Which class City function will provide the stored data when programs needs to get information of the City. These class City, class Distance, class Path will be processed by class Travel. Function of the class Travel is to calculate distance and heuristic, also gives the value to another class when needed, and handle back-end service for the user. Between them placed TravelController to handle front-end service of the user.

Population class holds multiple class Travel, in order for Genetic class to processed the Population class with using bio-inspired operators and Genetic rules for processing data. Genetic class also worked as bridge between Travel class and Population class to be accessed.