Amanda Shohdy, Jasmine Rodriguez, Edwin Rodriguez

amandashohdy@csu.fullerton.edu

jasminerrodriguez@csu.fullerton.edu

edwin.rodriguez@csu.fullerton.edu

Project 1 Report

Algorithm 1

Algorithm 1: Connecting Pairs of Persons was intended to sort a list of individuals based on their indices into couples. The first couple is presented as (0, 1), the second as (2, 3), and so on. This algorithm takes in any number of individuals and returns the minimum number of swaps that ensure each couple is sitting side by side. A swap occurs when choosing any two individuals and ensuring they switch seats (swap indices).

The assumptions of this algorithm are that n is less than or equal to 30 and even and that all elements in the row are unique. A variable is created to store the number of swaps that occur during execution. A vector is declared to store the user inputs of each individual in the row. A for loop is used to check through each individual in the row. If an individual is an even number, the following individual (individual i+1) should be the current individual + 1. If so, the next couple is evaluated. If not, the rest of the row is searched for the desired individual. Upon finding the desired individual, individual i+1 and the desired individual are swapped and the number of swaps is incremented. If an individual is an odd number, the following individual (individual i+1) should be the current individual – 1. If so, the next couple is evaluated. If not, the rest of the row is searched for the desired individual. Upon finding the desired individual, individual i+1 and the desired individual are swapped and the number of swaps is incremented.

The step count analysis of the efficiency class is determined to be $O(4n^2 + 6n + 5)$. This belongs to $O(n^2)$. The first while loop to collect user input is of time complexity 3n. The following for loop contains an if else statement. This outer for loop not included the nested for loop has time complexity 3n. The if and else statements have the same worst case time complexity; our analysis is based on the if statement. The following 2 if statements each have constant time complexity. The nested for loop has time complexity 4n. Thus, this section of code has time complexity $4n^2 + 3n$. Lastly, there are 5 lines with constant time complexity. This results in the time complexity of the algorithm to be $4n^2 + 6n + 5$.

Algorithm 2

Algorithm 2: Greedy Approach to Hamiltonian Problem was intended to pick the perfect starting city in a circular city road and each city has a certain amount of gallons of gas to offer. The assumptions are that the starting gas amount in the car is 0, the car will always have a positive amount of gas in its tank, and the miles per gallon are given.

The inputs to the function are 2 vectors that store the fuel and the city_distances values and the other input is an integer that measures the miles per hour. The algorithm will use two for loops; the first for loop will keep track of the starting city and the second for loop will loop through all of the cities. Before going through the second loop, the gas tank of the car is set to zero and the city is assumed to be correct. If the gas of the car is less than 0 when it arrives at the city, then the city is set to false and will exit from the loop. After the inner loop, the city is checked to see if it is correct. If it is correct, then it will return that city as the starting point.

The time complexity of the algorithm is O(n^2) because there are two for loops that have a set range and iterate through the entire list twice. Our algorithm consists of more statements other than the for loops but since that is the most time consuming, we recognize that as our worst case time complexity.