First Exam: Data Structures And Algorithms II

Ferry Loading II

Simón Marín Giraldo

School of Engineering
Informatics and Systems

Juan Guillermo Lalinde Pulido

Universidad EAFIT

Medellín

2020

1. Introduction

We were proposed to solve the Ferry Loading II problem using a greedy algorithm and also using a brute force or backtracking algorithm. This is an optimization problem in which you have to load a ferry with cars to transport them in the minimum time and trips quantity as possible. Basically, in the greedy solution we look for an optimal answer, but also in an optimal execution time.

2. Criteria for Greedy Algorithm

First of all, minimum quantity of trips done must be determined. In order to achieve this, number of cars is divided by the capacity of the ferry. With this, we get the minimum number of trips for exact division. In case this division has a modulus different from zero (not exact division) what we do is adding 1 to the minimum number of trips because we can't do a decimal quantity of trips. I decided to get this part of the solution with division because it gives us a constant time for each case. It works optimally, instead of a brute force solution in which you need to try all the possibilities and select the proper one. This minimum number of trips will be the total trips that will be done. After having this, what we do is getting the total number of trips. To achieve this, first we do is the operation m%n (where m is the number of cars and n is the capacity of the ferry). This operation is done for knowing how many cars will be taken into the first trip. After having this information and counting the first trip, what is done is counting the next trips with the ferry at its maximum capacity. This way, we get the same number of trips as the minimum previously calculated with m/n. Next, for the time, we set base cases in order to have an idea for the general case. For the time general case, what is done is that in every trip we add the time twice because that is what the ferry takes going to the other side and returning for picking more cars, unless we have only one trip left to to, in this case what is done is adding time, but only once because the ferry goes to the other side but doesn't need to return to the car terminal because all transportation job has been done.

3. Conclusions

After comparing the implementation in the greedy algorithm and the brute force algorithm, we see that the brute force solution works, but isn't as optimal as the greedy solution. In the greedy solution we use an array for saving the cars, which saves computation times, compared with the Linked List used in the brute force algorithm. Also, with basic operations such as division and modulus, we save plenty of computation time, instead of repeatedly and recursively trying values until finding the proper solution. While in the greedy solution we have a complexity of $O(n)$, in brute force we have a complexity of $O(2^n)$. Using greedy solutions, we can significantly reduce execution times. This is a problem with small data size. We must do things optimal for more complex problems with bigger data size.