ITF31519 - Portfolio, part 2, MDVRP

Tobias Hallingstad

December 22, 2021

1 Chromosome representation

1.1 My choice

I chose to represent the data using a normal Python class structure. I generate a class for `Depot`, `Vehicle` and `Costumer`. I store all the depots in a list, in main. The depots store all of the relevant information together with some lists. These lists contains all costumers, vehicles and the best vehicles. This means that there is no set way the data is defined.

Each depot have the correct amount of vehicles. The vehicles are assigned a route. The route is just a list of costumers in the order they are visited by the vehicle. The costumers are stored by reference in the route, this is so it is easy to get the costumer information from the list.

1.2 Other representations

Another way to represent the data is to store everything in a long list. Where the first elements are the number of costumers per vehicle. The rest are the ID of the costumer ordered in the route for that vehicle. This stores all of the routing information in one single list.

1.3 Argument for my representation

The way I represent the data is really easy to work with. Because all of the routes are stored in a class I have the possibility to do more checks when working with data. It is easy for me to add costumers to a route and I can use multiple methods to add them. Because of how python stores references to objects the size of the object is not very important, because i move the reference and not all the data.

2 Crossover and mutation is infeasible

2.1 Crossover

For crossover I choose to use a simple method to crossover the costumers form the vehicles. I select 2 vehicles (parents) then I run a loop for `X` number of iterations. Where each iteration tries to pick 2 costumers from each vehicle to be added to the other vehicle. The loop is used to try this some times, if one vehicle does not have space for the costumers. If the costumers fit, I can add them to the most optimal place in the route. If the costumers does not fit then I just leave them.

The code for this is:

```
for p1, p2 in parents:
    if random.uniform(0, 1) > crossoverRate:
        continue

    for _ in range(5):
        p1CC, p1CCI = selectRandomCostumer(p1)
        p2CC, p2CCI = selectRandomCostumer(p2)
        canAdd1 = canAddCostumersToRoute(p1CC, p2)
        canAdd2 = canAddCostumersToRoute(p2CC, p1)

        if canAdd1 and canAdd2:
            for c in p1CC:
                status = p2.addCostumerOptimal(c)
            for c in p2CC:
                status = p1.addCostumerOptimal(c)
        else:
            for i in range(len(p1CC)):
                p1.addCostumerIndex(p1CC[i], p1CCI[i])
            for i in range(len(p2CC)):
                p2.addCostumerIndex(p2CC[i], p2CCI[i])
```

2.2 Mutation

I did not get the time to add mutation. The method I planed on using is to randomly move one or two costumer from a random vehicle to another vehicle. There will be a test if the costumers can change. Another thing is to change what depot a border costumer is allocated to.

3 Parameter relationship

Because of how the code is written the parameters used for the GA are more a consequence of the functions created. This means that testing different parameters for the population size, generation number, crossover rate, and mutation rate was hard. I do have some observations on what my code does.

3.1 My observations

I have found that at the start of the code running the learning/improvements are large and many. After some time the improvements slow down and normalize close to a value. That value is the code normalizes around is in my case some distance off what is meant to be.

When the population size is larger the algorithm takes longer time. This is because of how I have written the crossover function. It takes longer to add a costumer when the costumer is added at the most optimal location. This function is not used when everything is initialized.

Because of time constraints I have not gotten the time to compleat the algorithm and put in the time to optimize things. This results in the code not working for all files. This is because of how i distribute the costumer.

4 Final things

I did not have time to compleat this task. With everything to read and understand and trying to write the code my self, the task was just to large for the time. I started with this task thinking that task 1 would take less time. Then I did task 1 to be done quick with that, but when that task took longer then expected I did not get the time to compleat this.

There are things that I would like to have added, like a mutation function. I also know that with a better crossover function combined with a mutation function the result will go way down. With each costumer assigned to its closes depot I do get some better result, but still not close to the required score.

4.1 Running the code

To get the result from each file just run the file. This will run 100 000 iterations for each file, unless there is some problems distorting the costumers, then the file is skipped (This is most normally file 6, 7, 10, 11). This is because there is a problem with adding all costumers, so I waited with fixing this and did not get the time to fix it. If the code should plot the route then line `577` and `578` needs to be commented out and correct file number needs to be added in the function on line `580`. This will also print the result in the format like the solution files, but not write it to a file (again because of time).

4.2 GitHub

I have used GitHub for version control. This means that the code is accessible on GitHub. The GitHub account the code is uploaded to is Tobhal (https://github.com/Tobhal) and is my own account.