Constraint Model

The situation we are asked to model is relatively complex, as we have many constraints and many of these values should be binary variables. Luckily, we are only finding a lower bound using linear programming. The main goal of this model is to figure out three objectives: which facilities to open, which vehicles to deploy for each facility, and which vehicles/facilities are responsible for serving each customer, while minimizing cost. Since we are using linear programming with continuous variables, there are some inconsistencies with the model that make it unrealistic, this will be discussed later.

Decision Variables

This problem is modeled using three collections of decision variables. First is `facility_open`, an array of values between 0 and 1 with size corresponding to the number of facilities. The value corresponds to how 'open' the facility is, which is a function of the number of vehicle it deploys and how much demand it is responsible for. Next is `num_vehicles`, an array of values between 0 and `numMaxVehiclePerFacility`, with size corresponding to the number of facilities. Finally, we have `customer_allocation`, a matrix of values between 0 and 1, of size `numCustomers` $*$ `numFacilities`. Each value in this matrix represents how much of a customer's demand is being met by a facility. For example, `customer_allocation[1][2]=0.3`, represents that the 30% of the 1st customer's demand is being met by the 2nd factory.

Constraints

The first constraint is regarding customer demand. For each facility, I make sure that customer demand does not exceed the capacity of the facility multiplied by how open it is. Customer demand for a facility is calculated by multiplying every customer's demand by the ratio that the factory is responsible for.
The second constraint ensures that driving distance required by any facility is less than the distance the facility is capable of. The driving distance required by any facility is calculated by multiplying the distance from the customer to the factory with the ratio that the factory is responsible for. The maximum driving distance of the facility is calculated by multiplying the `truckDistLimit` with the number of vehicles at that facility.
The third constraint ensures that the number of vehicles used at a facility is proportionally less than how open the facility is. This calculation is simply the number of vehicles being used at a facility over the maximum number of vehicles per facility.
The fourth constraint ensures that every customer's demand is perfectly met. We sum across rows of the `customer_allocation` matrix ensuring that this value is equal to one.
An implicit constraint is that the number of vehicles at any one facility is bound by the value of `numMaxVehiclePerFacility`, because the variables within `num_vehicle` cannot exceed this value.

Objective

Finally, the objective is the cost calculation. This consists of three parts, the opening cost for each facility, paying for each truck, and paying for each round trip. The opening cost for each

facility is simply the cost multiplied by how open it is. Then, paying for each truck is multiplying the `truckUsageCost` with the number of vehicles at each facility and summing this value. Paying for each roundtrip is multiplying corresponding values of the `allocCostCF` matrix with the `customer_allocation` matrix.

Oversights & Limitations

Since we are using linear programming, there are a few things that make this model impossible to implement in real life. First, we have fractions of vehicles, which obviously doesn't make sense. The number of vehicles is calculated by summing how much of one vehicle is needed for each trip from the factory. Another issue is that we do not explicitly calculate paths for each vehicle. We also have facilities partially open, which should be a binary variable. The handout also specifies that each customer should be served by one facility. We also fail to factor in that customers can be serviced during the same trip and a round trip is not necessary for every customer. However, we lack a variable to track capacity of a truck meaning that it is hard for us to know how many customers we can service during one trip.

Conclusion

This project was very interesting, because it required more thought and planning than anything else. The lines of code I wrote were extremely simple and extremely quick, but thinking about the most efficient data structures for the constraints was thought-provoking. It was a bit difficult for me to wrap my head around the idea of turning a problem with discrete variables into continuous variables, but I ended up writing the constraints as if the model used discrete variables then made slight transformations in order to make it suitable for continuous variables. I likely spent around 7 hours on this project.