Optimization
Project 2 – Integer Programming

Deliverables

One Python code file (.py or .ipynb) and one well-written PDF file, submitted to Canvas. Your report should go into some detail about how you solved the problem, include some graphs that explain your results, and include relevant code chunks in the final output. 60% of your grade will be based on whether you get the problem right or not, the remaining 40% will be based on the quality of your analysis. We will re-run your code with a new data set. If you don't get the right answer or your Python file doesn't run, we will go through your code and give partial credit accordingly. The easier it is to read your code the easier it is for us to understand what you're doing, so use a lot of comments in your code!

Problem Overview

The travelling salesperson problem (TSP) is a difficult integer program to solve. The solution that we discussed in class is actually not a very efficient way to solve it when the number of nodes increases. When the number of nodes is quite large it is almost impossible to find the truly optimal path. In this project we'll work on a variation of the TSP that is faced by logistics companies every day. In a city, there is one central distribution center for package delivery. All packages are loaded onto trucks at the distribution center, then the trucks deliver the packages, and come back to the distribution center at the end of the day. We start with simply the locations of all packages that need to be delivered, and then we must decide onto which truck to each package should be loaded, and once the trucks are loaded what is the optimal route that starts and stops at the distribution center for each truck.

This problem can be formulated almost exactly like we formulated the standard TSP in class, with a couple small modifications. We still have the constraint that no loops are allowed except for loops that go through the distribution center, and then we can change the number of links into and out of the distribution center to be equal to the number of trucks. I formulated a problem that has 50 drop-off locations,1 distribution center, and 3 trucks; it took gurobi more than 2 hours to solve this problem. The focus of this problem will be on approximation algorithms to solve this logistics problem.

Simulated Annealing

One approximation algorithm to solve the standard TSP is to initialize with a randomly chosen path. Then we go through and repeatedly look at possibly changing portions of the path. There are two types of changes considered: reverse and transport. For a reverse change you randomly pick 2 nodes, 1 is labeled the starting node, and 1 is labeled the ending node; let's say these nodes are i and j. We find the segment of the path that starts at node i and goes to node j, disconnect it from the entire path, reverse it and put it back into the path. For example, if we had a 6-node problem and we have a path that goes 0 -> 1 -> 2 -> 3 -> 4 -> 5 -> 0, and randomly picked the starting node to be node 5 and the ending node to be 2, then the segment of the path that connects 5 to 2 is 5 -> 0 -> 1 -> 2. To reverse this path, we find the node that used to go into 5, and make it go into 2, and find the node that used to come after 2 and make it come after 5, so now the new path after reversing would 4 -> 2 -> 1 -> 0 -> 5 -> 3 -> 4. The other type of change to the path, transport, similarly finds a random start and end node and finds the path between those 2 nodes. Then a transport change removes that sub-path from the loop, closes the loop with the remaining nodes, finds a random link on new loop, splits the loop there and pastes the sub-path. Going back our earlier example, we would remove the 5 -> 0 -> 1 -> 2 sub-path, and close the loop

on the remaining nodes, to get 3 -> 4 -> 3. We then choose one of these links at random to cut, let's say 3 -> 4, and put the sub-path in where that link was, to get 3 -> 5 -> 0 -> 1 -> 2 -> 4 -> 3.

With these two types of changes, we can now put them together to describe the simulated annealing policy. 1) Initialize a loop completely at random. 2) Toss a fair coin and pick randomly between a reverse and transport change. 3) Try the change. 4) If it reduces the total distance travelled keep the change and go back to 2; if the change does not decrease the distance travelled don't keep it and go back to 2. As described, this is a greedy algorithm and may get stuck in a local minimum. An easy way to fix this issue is to randomly accept changes sometimes even if they increase the distance. When a change increases the distance, instead of not accepting it we can toss a weighted coin (more on the weight to come), and if it comes up heads keep the change even though it leads to an increase in distance travelled. We don't want the probability of accepting very bad changes to be very high. We also want the probability of accepting bad changes to get smaller as the algorithm progresses. To accomplish these goals, pick some initial value of $\epsilon \leq 1$. For each change that increases the distance call that increase in path distances d, then set the probability of accepting a bad change equal to $\epsilon/\exp(d)$. Then after each iteration of the algorithm, shrink ϵ a little bit, perhaps by multiplying it by 0.99. Then after many iterations of this algorithm, you should have a pretty good path. This is a heuristic algorithm and is NOT guaranteed to converge to the optimal path, but typically does a pretty good job.

Multiple Trucks

If there are too many nodes to solve the basic TSP, then there may also too many nodes to solve to multi-truck TSP. One heuristics strategy for this is to cluster the locations of all the deliveries into as many clusters as there are trucks, using something like k-means or hierarchical clustering. Or some other clustering method, like splitting the x-y plane into wedges with equal number of points in each wedge. Then once the stops are clustered, we solve a standard TSP on each individual cluster (adding the distribution center to each cluster). We may be able solve each individual TSP exactly with gurobi, or we may have to use simulated annealing on each cluster's TSP, depending on the number of stops in the clusters.

Specifics

1) Attached to this project description is a csv file with the x-y locations of 500 packages that need to be delivered today. The distribution center is located at the origin: (0,0). The distance between any 2 points is calculated using the standard Euclidean distance: $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. Solve the problem that minimizes the total distance travelled using all 10 trucks. You may use whichever algorithm you want to solve this problem.
2) For every mile travelled delivering packages it costs $1 for gasoline, labor, truck depreciation and so on. Additionally, every driver that works today gets an additional payment of $300, no matter how many packages they deliver. How many trucks should you use to make your deliveries? To answer this question, pose it as an optimization problem and solve it.
    a. I have not solved this problem given this dataset, so I don't know if these costs are reasonable…but give it a shot anyway. If some other costs look like they'll make more sense, feel free to use those, but justify why you're changing costs.
3) Tomorrow there is a new set of packages that need to be delivered to a new set of locations. The locations are in the second csv file. You want to have some sort of consistency in the neighborhoods/routes that each truck travels to, so you're thinking about putting packages that

need to be delivered close to yesterday on the same truck as yesterday, and then solving the TSP for each truck. How much does this increase the total cost relative to completely starting over to solve the problem from scratch? Should you have consistency in the routes, or is worth it to start from scratch every day? Should there be some penalty/cost for drastically changing routes on drivers?

4) Do you have any other ideas for another way we could split up the nodes? Can you come up with a way to incorporate the truck selection into the simulated annealing process? Maybe randomly switch packages from one truck to another? Be creative!
5) You work for a logistics company. Your boss has been manually sorting packages and scheduling routes. Your boss is tired of doing this and knows that you took a class that covers the TSP. Could you decrease costs by using optimization to plan delivery routes? Describe the advantages and disadvantages of this. Your boss is pretty technical and understands optimization, so don't be afraid to include quantitative material. Your boss is also busy, so be sure to include some visualizations to get the important points across. For the purpose of your report, you can assume that your boss is interested in the data posted with the project. Write this as if you were going to give it to your boss as a technical report that may be shared with the CEO. That means, make it look GOOD!