Local Search:

The story of how we did a lot of work just to find out that we didn't have to.

Kate Nelson and Anirudh Narsipur

Initial Solution

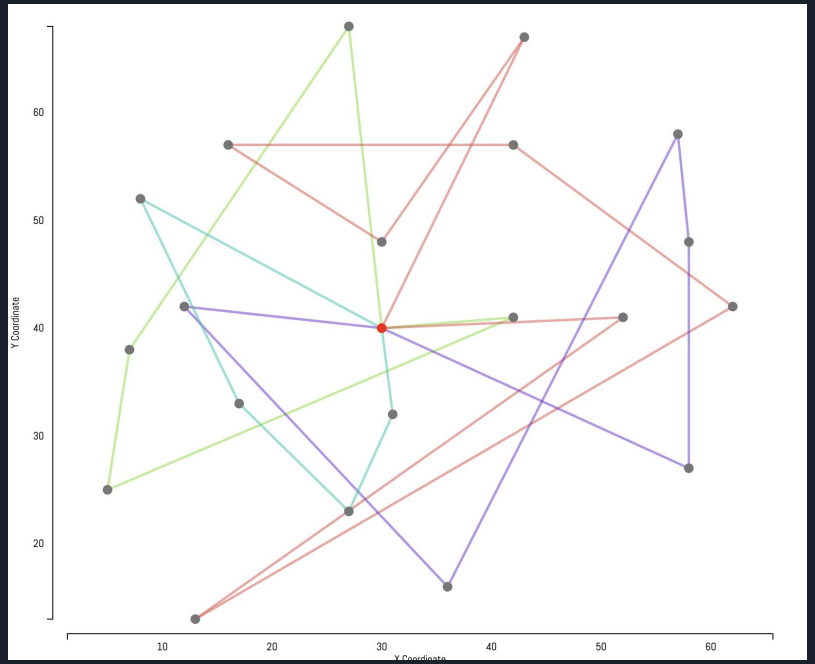
- Get an IP solver to solve the feasibility problem.
- Read initial solution left to right: Eg: If Vehicle 2 has been assigned to customers 2, 5, and 8, then the route is : Depot -> 2 -> 5 -> 8 -> Depot.
- 2-opt each route
- Can do greedy heuristics, but that's hard and takes a lot of time…

Solve The IP Feasibility Problem :
V : Vehicles , C : Customers , Demand Vector D
Variables : VxC matrix : $M_{V,C}$
$V + C$ Constraints:

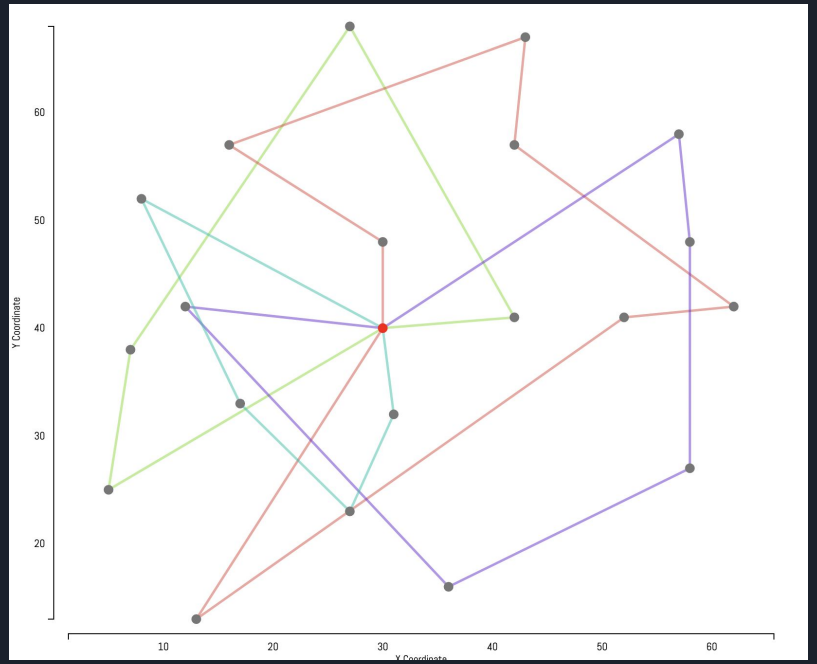
$$\sum_{i=0}^{c} M[:, i] = 1$$

$$\sum_{j=0}^{c} M[k, j] * D[j] <= \text{capacity}, \forall k = 1 : V$$

Initial Solution



Post 2-Opt Solution

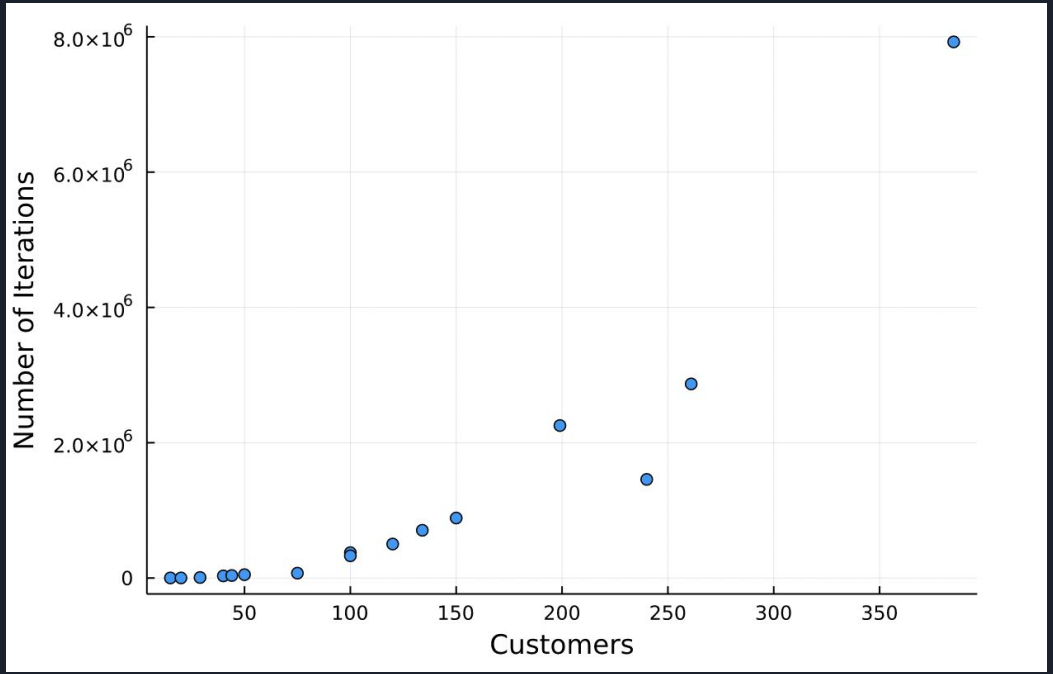


Local Search

- Simulated Annealing
- Pick two random nodes (customers)
- Calculate new objective value by swapping their positions and checking the distance. If the swap leads to an improvement, then accept and conduct the swap.
- If the swap produces a worse objective, accept the worse move with probability $e^{(s-s')/T}$; otherwise, reject the swap.
- Swapping is completely random, so do multiple times (n = 30).
- Pro: very low overhead (~0 memory allocation). Largest instance takes ~3 seconds.

Tune Hyperparameters

- First, tune temperature such that the first improvement is found fast
- Temperature = (initialObjective / ln(0.97)) * 0.001
- How many iterations required for convergence ?
  - Run for large number of iterations (10^4 * C)
  - Check when the last improvement was made
  - Relationship is approximately quadratic.
- numItr = 154579.1256 + 150.2166590*x + 54.10538*x^2



Failings

- Neighborhood Based Methods
  - K-Means
  - Greedy
  - Rebuild based on neighborhood

- Solver Based Methods (CP,LP) -> Too Slow
- More complicated LocalSearch variants

Time + Takeaways

- Spent about 50 hours over a month
- Optimized version of simple solution worked best
- Could have spent a few years exploring all possible heuristics/solution methods