Designing the Subproblem Algorithm

Summary: After many attempts to solve the subproblem using purely Bellman-Ford, I have found that each graph model has unfixable limitations. Specifically, it seems like one cannot use a pure Bellman-Ford graph implementation and design it in such a way that time and motion constraints (e.g. time windows constraints on y^S and y^E hold, and vehicles do not go back to a job location they were at previously) hold.

I'd like to receive guidance on how to, in the subproblem, successfully generate feasible routes with most negative reduced cost that also satisfy constraints put in the RMP. Is this achievable with purely Bellman-Ford? Is there a mix of both Gurobi/post-Bellman-Ford checking that I can use?

Introduction:
Assume we have a set of initial routes $\mathcal{Q}_0$ which we used to solve the restricted master problem. The way I picture it is: I would like to generate a better route (column) $q' \notin \mathcal{Q}_0$ which deliberately violates the criterion of the dual that

$\sum_{i \in \mathcal{I}} \pi_i a_i^q + \rho + \sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}} \mu_{it} \delta_{it}^q \leq C^q$ FOR EACH $q \in \mathcal{Q}_0$. That means we seek a route q' such that

$C^{q'} - \sum_{i \in \mathcal{I}} \pi_i a_i^{q'} - \rho - \sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}} \mu_{it} \delta_{it}^{q'} < 0$ (reduced cost is strictly negative).
Here q' can be anything, as long as $q' \in \mathcal{Q}$. Note by strong duality that $q' \notin \mathcal{Q}_0$.

We have already created the entire possible route network. (This was done earlier in the notebook.) Then, feeding in this route network, with weights and all, Bellman-Ford can solve the implementation.

The goal for the subproblem is to avoid having to go through all feasible routes and searching for the best solution. Instead, we can find the best route with the most negative reduced cost through Bellman-Ford, which can handle negative indices. Bellman-Ford is must faster than Gurobi.

After doing so, we can build our graph.

Attempts at Modeling

Attempt 1: I have attempted to create a graph with nodes $[i]$ (so there's 27 nodes total: 25 job nodes, one start depot, and one end depot). However, this design fails to include the dual variables which require both i and t, namely in the summation $\sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}} \mu_{it} \delta_{it}^q$. Therefore, we are forced to have separate nodes for the same i but different t, because the graph weights μ_{it} might be different between different t for the same i.

In the vehicle routing lecture (15.083 lecture 20), the scenario in which vehicles must run routes that altogether cover all location nodes is solvable using column generation and a shortest-path subproblem, because the subproblem can be done with graph nodes

i (there is no need to consider time windows, nor is there a dual variable in i and t). Furthermore, all restrictions on vehicle movement (e.g. no going back to a node you've already visited) are built-in to the structure of the graph: Bellman-Ford eliminates negative weight cycles.

Attempt 2: Because we cannot create a graph with nodes in i, we must do so by considering both i and t. This resembles Figure 1. Here, I circumvented the issue of "a vehicle can leave the depot at any time, not just 0; it can return to the depot at any time, not just 51" by creating a "superstart" node with arrows into each $[0,t]$ node with weight 0, and a "superend" node receiving arrows of weight 0 from each $[26,t]$ node. Then run Bellman-Ford from the superstart node to the superend node!

But this implementation cannot handle the requirement that a vehicle not return to a job location it already went to before (whereas the nodes only in i did so in Attempt 1). For this reason, optimal routes of the subproblem consistently bounced between e.g. two close location nodes to repeatedly "collect" the large positive numbers π_i that are gained when one enters job location i, causing $-\sum \pi_i$ to be very negative (while the increase in route distance was sufficiently small). There was no way to prevent this behavior.

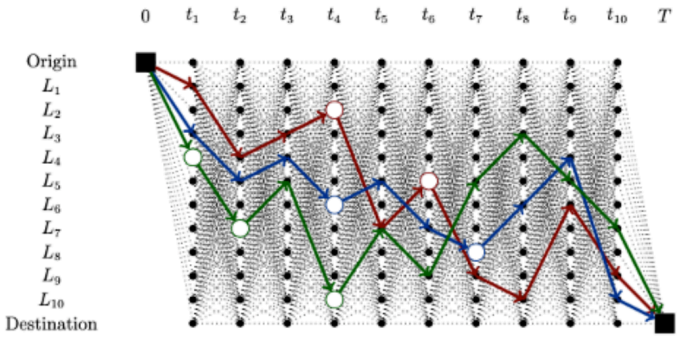

Figure 1. A node graph in both i and t, slide 23 of 15.083 lecture 20.

I see two ways to resolve this issue.
1) We enforce directly (somehow) in the subproblem that routes cannot be at a job location, get out of it, and then come back.
2) We design the i,t graph in such a way that no extra π_i will be collected as reward if a route did perform in-and-out-and-back-in behavior (but I do not see a way to do this).

Questions:
1) Is there a way to use only a Bellman-Ford algorithm (as desired) to encapsulate both the subproblem equation $C^{q'} - \sum_{i \in \mathcal{I}} \pi_i u_i^{q'} - \rho - \sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}} \mu_{it} \delta_{it}^{q'} < 0$ as well as the required constraints on q, y^S, y^E?

2) How would we incorporate y^S and y^E and/or time window constraints into a graph algorithm which seems to only be able to deal with q?