CSC4120 Project Report

121090454 Yaqi QIAO   121090620 Yueyan WU   121090002 Yuxuan BAI

Question3

In this question, we try to use Integer Linear Programming to solve the PTP problem.

As a preparation work, we calculate each vertices pair's shortest distance by directly calling `nx.all_pairs_dijkstra_path_length` function.

Then we construct a relative integer linear programming:

Step1. We construct the decision variables:

$$\begin{gathered} X[i,j],(i,j)\in edges : 0\ or\ 1, 1\ means\ the\ car\ pass\ this\ edge. \\ Y[h,n], h\in H\ and\ n\in Nodes : 0\ or\ 1, 1\ means\ the\ friend\ walk\ from\ h\ to\ pickup\ place\ n. \\ R[i,j],(i,j)\in edges : int,\ R\ means\ residual\ flow. \end{gathered} \tag{1}$$

Step2. We construct the objective function

$$\min \alpha \sum X[u,v]G[u][v] + \sum Y[h,n]dis[i][j] \tag{2}$$

Step3. We construct the subjective functions

It's easy to construct the constraints for X and Y:

To make sure car's path construct a circle: $\sum_{j\in pre} X_{ji} = \sum_{j\in suc} X_{ij},\ for\ i\ connecting\ edges$

To make sure each friend goto one pick-up place: $\sum_{n\in Nodes} Y[h,n] = 1$

Importantly, to make sure there is only one path in G (the one starting and ending at v_0, we use a concept "residual flow" to control.

$For\ each\ node\ except\ v_0 : \sum_{j\in suc} R_{ij} - \sum_{j\in pre} R_{ji} = \sum_{h\in H} Y[j,i]$

$At\ v_0,\ \sum_{j\in suc} R_{0j} = \sum_{h\in H} Y[h,0], \sum_{j\in pre} R_{j0} = size\ of\ H$

$For\ each\ edges, X[i,j] * size\ of\ H\ \geq R[u,v]$

Then we got the answer with some $X[i,j] = 1$, we simply do a dfs to reconstruct the answer path.

To implement the integer linear programming, we use solver `cbc` and `pulp`. And we use hyperparameter `sec` to control the program finish in 1min. Before you run the code please make sure you have download these tools:

```
pip install cbc
pip install pulp
```

The result is shown below, more details can be found in `.out` files.

Test	Cost	Runtime(s)	Test	Cost	Runtime(s)
1	3.334	0.96	6	252.899	0.995
2	69.0	1.487	7	714.0	1.554
3	138	2.468	8	1782.299	58.892
4	130.2	2.634	9	5770	58.962
5	255.0	17.65	10	11775	27.429

Question 5

Question 5.1

First, we compute the shortest path between any of the two friends' house and the shortest distance from my house to each of my friend's home. Then, we extract the vertices of my house and my friends' house and connect them with an edge which has a weight of the shortest path between and operate the TSP problem on the graph(notice that the triangle inequality still holds in the new graph), and we get the solution of PHP. Therefore, we can reduce the PHP problem to the metric TSP problem. Since by lecture, the M-TSP problem is NP-hard, so PHP is NP-hard.

As we have already known that PHP is a NP-hard problem, we want to reduce PHP to PTP problem, which means PTP is a harder problem, so PTP is also NP-hard.

We consider the case when $\alpha \leq \frac{1}{2}$. Under this condition, the solution of a PTP problem must not contain a path (h_i, p_i) that one friend walk from home to the pickup location. Instead, the car will reach every friend's home to pick them to minimize the formula

$$\alpha \sum_{i=1}^{n} w_{u_{i-1}u_i} + \sum_{m=0}^{|F|-1} d_{p_m h_m} \tag{3}$$

We can prove this by contradiction:

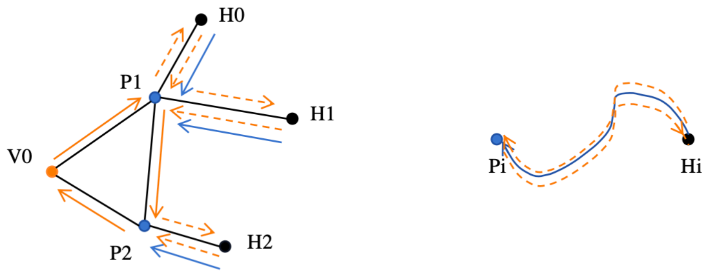
Assume in the PTP solution, there is one path (h_i, p_i) that one friend walk from home h_i to one pickup location p_i. As now $\alpha \leq \frac{1}{2}$, we can let the car drive from p_i to h_i then go back to p_i along the same path. Now we have:

$$\alpha(w_{p_i h_i} + w_{h_i p_i}) \leq d_{p_i h_i} \tag{4}$$

That is, we construct another solution with smaller cost:

$$Cost_{new} = \alpha(\sum_{i=1}^{n} w_{u_{i-1}u_i} + w_{p_i h_i} + w_{h_i p_i}) + \sum_{m=0}^{|F|-1} d_{p_m h_m} - d_{p_i h_i} \leq \alpha \sum_{i=1}^{n} w_{u_{i-1}u_i} + \sum_{m=0}^{|F|-1} d_{p_m h_m} = Cost_{old}$$

which is in contradiction with that the PTP solution minimize the formula. The below figure shows one example.



Because the car will reach every friend's home to pick them in the solution of a PTP problem with $\alpha \leq \frac{1}{2}$, it is actually a solution of a PHP problem. So, for a PHP problem, we can transfer it into a PTP problem with $\alpha \leq \frac{1}{2}$. The transformation is in polynomial time as the only operation is adding α. Then the PTP solution is directly the PHP solution. Now we finish the proof: $PHP \leq pPTP$, PTP is NP-hard as PHP is NP-hard.

Question 5.2

We assume that $\alpha = 1$. Consider a car path p' that: car drives along the PTP solution's car path, and every time the car reaches a pickup location p_i, the car will follow friend's path to reach h_i and go back to p_i for each (h_i, p_i) (for one p_i here may be several h_i, the car will repeats the process with the corresponding times). Because the PHP solution is the minimum one of car paths starting from v_0, reaching every h_i, and finally reaching v_0 again, which include p', obviously we have:

$$Cost(PHP) \leq Cost(p') \tag{6}$$

Then we consider the numerical relation between p' and PTP solution. With $\alpha = 1$, for each $w_{u_{i-1}u_i}$, representing the car path, the cost is the same in p' and PTP solution; for each $d_{p_m h_m}$, the cost of p' is twice of PTP solution as a friend in PTP will walk along the path once and a car in p' will walk along the path twice. That is:

$$Cost(p') = \sum_{i=1}^{n} w_{u_{i-1}u_i} + 2\sum_{m=0}^{|F|-1} d_{p_m h_m} \leq 2(\sum_{i=1}^{n} w_{u_{i-1}u_i} + \sum_{m=0}^{|F|-1} d_{p_m h_m}) = 2Cost(PTP) \tag{7}$$

Then we have:
$Cost(PHP) \leq Cost(p') \leq 2Cost(PTP)$,
so:

$$\beta = \frac{C_{php}}{C_{ptpopt}} \leq 2 \tag{8}$$

The bound is tight as we can provide an instance where $\beta = 2$ as the graph shown below. In this graph, $Cost(PHP) = 2Cost(PTP)$: