[1].Fiber Connectivity

Let, G = (V, E)

Where, G is graph, V is the set of vertices, and E is the set of edges. In the problem, V = C (the set of the cities) and E = R (the set of the roads). I have an edge e : (u,v) between two nodes u and v in the graph G only if we have a road between the corresponding cities u and v in the original problem. I rephrase the question as: given this graph G, is there a subset of edges $T \in E$ such that $|T| < |V| - 1$ and $1 \leq d_T(u) \leq k$ for all nodes $u \in V$ and all nodes in V are connected together.

1.NP Membership

I will prove that Fiber connectivity belongs to the class NP. Let T is a candidate solution, which must be solved in polynomial time. T is a spanning tree, and I must check:

- $|V| - 1$ edges are used, all nodes are connected.
- all vertices have degree less than or equal to k.

I will try to run BFS, or DFS algorithm to check if it is a tree or not and see if I have a loop or not. This procedure takes polynomial time $|V| + |E|$.

Now I must check if T connects all vertices. I can check whether the number of edges of T is equal to $|V| - 1$ or not. Any tree with $|V| - 1$ edges must have V nodes.

Finally, I can check that all vertices in T have degree less than or equal to k by examining each vertex one by one. I can check if the candidate solution is a right solution or not in polynomial time in $|V| + |E|$

2.Reduction

Now, I show the problem is NPC by reducing the Hamiltonian Path which is NPC problem. Given an instance of Hamiltonian Path, I will convert it to a same instance of Fiber Connectivity in a polynomial time. In Hamiltonian Path, I am asked to check whether it has a simple path that visits all vertices in given graph G. The most important one for the reduction is that Hamiltonian path is a spanning tree with all degrees of vertices smaller than or equal to 2. Thus, it is trivial to transform a Hamiltonian path instance to a Fiber Connectivity instance. Such as using the same graph (G) for the original Hamiltonian Path, and set the degree bound k = 2

a.Polynomial time of reduction

I will just use the same graph (G), thus, it is trivial.

b.Correctness of reduction

Hamiltonian Path will be in Yes-instance iff the corresponding Fiber Connectivity is a Yes-instance.

(→): If the Hamiltonian Path instance is a Yes-instance, it has a path P that visits all vertices. All paths are also a tree without loop. Thus, P is a tree which visits every vertex, for example, P is a spanning tree. Besides, the start and end vertices have degree 1 and all others have degree 2, because P is a path, thus all vertices have degree at most 2. As a result, P is a solution for the

Fiber connectivity instance.
(←): If the Fiber Connectivity instance has a solution T, then I will have a polynomial time algorithm to solve Hamiltonian Path using the reduction. Since Hamiltonian Path is an NPC problem, thus Fiber connectivity is also NPC.

[2].Minimum Set Cover

1.Branch and Bound Set Cover

The partial solutions will be the form $X_1, X_2, \dots, X_n$:

$$X_i = \begin{cases} 0 & if\ S_i\ is\ not\ included \\ 1 & if\ S_i\ is\ included \\ None & if\ it\ is\ not\ decided\ yet \end{cases}$$

And all initial partial solutions are undecided form $X_i =$ None

(a).What is a subproblem?

The subsets which is selected as inclusion (X[i]=1) in the SET COVER is:

$$\hat{S} = \{S_j | X_j = 1, j = 1, \dots, n\}$$

The set of elements covered by subsets which is selected in the partial solution is:

$$\widehat{U} = \bigcup_{S_j \in \hat{S}} S_j$$

The corresponding subproblem will be a smaller instance of the SET COVER problem with universe $\overline{U} = U \backslash \widehat{U}$ adjusting by removing the elements covered by at least one subset which is selected in the partial solution. And I make a set of subsets derived from the original subsets with X[i] = None, $\bar{S} = \{S_j \backslash \widehat{U} | X_j = None, j = 1, \dots, n\}$

(b). How do you choose a subproblem to expand?

I can select the subproblem that has the smallest number of elements with minimal $|\widehat{U}|$.

(c). How do you expand a subproblem?

I expand a subproblem by choosing all elements i with X[i] = None, and derive from each two sub-problems corresponding to setting X[i] = 0 and X[i] = 1 respectively.

(d). What is appropriate lower bound?

An appropriate lower bound for a partial solution X' is:

$$L_{X'} = \sum_{j=1,\dots i | X_j = 1} 1$$

2.A simple greedy heuristic

I assume the SET COVER instance is feasible, for example, there is some subset of subsets S_i that can cover all elements in the universe. The idea is that at each iteration we will choose to add to our SET COVER the subset that covers the biggest number of remaining uncovered elements from the universe.

Input: a set of elements $\mathrm{U} = \{u_1, u_2, \ldots, u_m\}, a\ set\ of\ subsets\ S = \{S_1, S_2, \ldots, S_n\}$

```
Output: a feasible solution $\mathrm{S}' \in \emptyset$
Set $\mathrm{S}' \leftarrow \emptyset$
Set $\mathrm{U}' \leftarrow U$
While $\mathrm{U}' \neq \emptyset$ do
        $S_{add} \leftarrow argmax_{S_i \in S \backslash S'} |S_i \cap U'|$
        $\mathrm{S}' \leftarrow \mathrm{S}' \cup S_{add}$
        $\mathrm{U}' \leftarrow \mathrm{U}' \backslash S_{add}$
```

Complexity

(1). $S_{add} \leftarrow argmax_{S_i \in S \backslash S'} |S_i \cap U'|$: O(mn)

(2). $\mathrm{S}' \leftarrow \mathrm{S}' \cup S_{add}$, $\mathrm{U}' \leftarrow \mathrm{U}' \backslash S_{add}$: work liner in m and n

(3). While loop: min(m,n)

Thus, total complexity is O(min(m,n)*mn) or $\mathrm{O}(m^2 n)$

3.Local Search

(a). What could be a possible scoring function for such candidate solutions?

Let S' be a candidate solution. Then the scoring function can be:

$$\mathrm{g}(S') = \frac{\left|(\bigcup_{S_j \in S'} S_j) \cap U\right|}{|S'|}$$

That is, the score of a candidate solution is the ratio of the number of elements of the universe that its subsets cover, to the number of subsets it includes. Such a scoring function favors solution that cover many elements with only a few subsets.

(b). What would be a Neighborhood (or Moves) you would consider using for your local search to move from one candidate solution to other 'nearby' solutions? How many potential neighbors can a candidate solution have under your Neighborhood (using Big-Oh)?

One neighborhood I can adopt is that of the candidate solutions that differ by exactly one subset, in that we either add a new subset to the current solution, or I remove one of the subsets already included in our current solution. Since there are n subsets in total and each is either considered for addition or removal in relation to the current solution, the size of the Neighborhood is O(n).

(c). Why would you consider adding Tabu Memory and what would be remembered in your Tabu Memory?

Tabu Memory is useful in avoiding local minimum by preventing repeated solutions. One implementation of Tabu Memory would remember the k last subsets that were added or removed in the latest k solutions. This make the algorithm explore new options by selecting solutions which include subsets that were not recently explored.

[3].Optimizing Amazon's Operations

Set of n possible warehouse (1-n) to serve m possible cities (1-m). Given information is fixed construction cost and minimum distance to the cities. Decision is whether to build (X[i] = 1) or not build (X[i] = 0) each warehouse. Objective is to minimize fixed plus distance.

1. Branch and Bound

(a). What is the sub problem?

The partial solutions will be the form $X_1, X_2, \dots, X_n$:

$$X_i = \begin{cases} 1 & \quad if\ Decision\ is\ whether\ to\ build\ warehouse \\ -1 & \quad if\ Decision\ is\ whether\ not\ to\ build\ warehouse \\ 0 & \quad Unknown \end{cases}$$

The subsets which is selected as built (X[i] = 1) is:

$$\mathrm{F} = \{f_j | X_j = 1, j = 1, \dots, n\}$$

Then, I assume $\mathrm{F} = \{f_1, \dots, f_i, \dots, f_k\}, k\ is\ less\ than\ n$

The given equation which is selected in the partial solution is:

$$\mathrm{A} = \sum_{i \in \mathrm{F}} f_i + \sum_{j \in \{1,\dots,m\}} min_{i \in F} d(i, j)$$

(b). How do you choose a subproblem to expand?

I can select the warehouse location that minimize the equation $\mathrm{A} = \sum_{i \in \mathrm{F}} f_i + \sum_{j \in \{1,\dots,m\}} min_{i \in F} d(i, j)$

(c). How do you expand a subproblem?

I expand a subproblem by choosing all elements i with X[i] = None and derive from each two sub-problems corresponding to setting X[i] = 1 and X[i] = -1 respectively.

(d). What is appropriate lower bound?

Minimum transport cost over all built (X[i] = 1) or unknown. Plus, fixed cost of built.

2.Outline a simple greedy heuristic for the problem

Let the facility location instance consist of cities D, facilities F, metric d on $\mathrm{D} \cup \mathrm{F}$, and construction cost $f_i \in \mathbb{R}_+$ for each $\mathrm{i} \in F$. Then the goal is to open a set $F^* \in F$ of facilities that minimizes $\sum_{i \in F^*} f_i + \sum_{j \in D} d(j, F^*)$. Above for any city $\mathrm{j} \in D$, $d(j, F^*) = min_{i \in F^*} d(j, i)$ denotes the distance form j to its nearest facility in F^*.

The facility location problem can be cast as a set covering problem where elements are the cities D, and sets correspond to 'stars' centered at some facility. Any set can be represented as (i, A) where $\mathrm{i} \in F$ is a facility and $\mathrm{A} \in D$ is a subset of clients; the cost of this set is $f_i + \sum_{j \in A} d(i, \mathrm{j})$ The modified greedy algorithm is as follows.

```
Set $F' \leftarrow \emptyset$
While $D \neq \emptyset$ do:
    1. Pick a set (i, A) where $i \in F$ and $A \in D$ that minimizes the ratio $\frac{f_i+\sum_{j\in A} d(i,j)}{|A|}$
    2. Set $F' \leftarrow F' \cup \{i\}$, $f_i \leftarrow 0$, and $D \leftarrow D \backslash A$
Output: F' as the set of constructed facility
```

Time Complexity

O(min(m,n)*mn) or $O(m^2n)$

3.Local Search

(a). What could be a possible scoring function for such candidate solutions?

Let F’ be a candidate solution’s. Then possible scoring function is:

$$g(A') = \frac{\sum_{i\in F'} f_i + \sum_{j\in\{1,\ldots,m\}} min_{i\in F'} d(i,j)}{|F'|}$$

(b). What would be a Neighborhood (or Moves) you would consider using for your local search to move from one candidate solution to other ’nearby’ solutions? How many potential neighbors can a candidate solution have under your Neighborhood (using Big-Oh)?

Iteratively add or remove a location at a time, add a location to set W if it has a low building cost but can reduce the sum of distances to cities. Remove a location from W if it has a high building cost but can be replaced by other locations in term of distances. The number of neighbour solutions is O(n).

(c). Why would you consider adding Tabu Memory and what would be remembered in your Tabu Memory?

Tabu Memory is useful in avoiding local minimum by preventing repeated solutions. One implementation of Tabu Memory would remember the k last subsets that were added or removed in the latest k solutions. This make the algorithm explore new options by selecting solutions which include subsets that were not recently explored.

Reference

Problem1: https://www.geeksforgeeks.org/proof-hamiltonian-path-np-complete/
Problem2: https://www.geeksforgeeks.org/set-cover-problem-set-1-greedy-approximate-algorithm/

Problem3: https://ocw.mit.edu/courses/civil-and-environmental-engineering/1-204-computer-algorithms-in-systems-engineering-spring-2010/lecture-notes/MIT1_204S10_lec17.pdf