CS 577: HW 6

Daniel Ko

Summer 2020

§1 Give an efficient algorithm to decide if this is possible, and if so, to actually choose an ad to show each user.

§1.1 Construct a network flow model for this problem. Clearly state the meaning of each component (node, edge, capacity) of the flow network that you construct, present your algorithm in pseudocode, and give the computing complexity analysis of your algorithm.

We construct a graph $G = (V, E)$ to model the network flow.

§1.1.1 Components for graph

a. **Nodes**
   - Start with a source node s.
   - Add a column of n nodes, denoted as u, to represent the number of users.
   - Add a column of k nodes, denoted as dg, to represent the number of demographic groups.
   - Add a column of m nodes, denoted as a, to represent the number of advertisers.
   - End with a sink node t.

b. **Edges and capacities**
   - Add an edge from source node s to each user node. Let the capacity of these edges be 1 to represent that we only want 1 ad for a user.
   - Add edges from each user node to the corresponding demographic groups it belongs to. Let the capacity of these edges be 1 to represent that only 1 ad for a particular demographic groups can be shown to a user.
   - Add edges from each demographic groups to each advertiser that wants to show its ads to. Let the capacity of these edges be r_i, as we only care about the lower bound of the number of users to show ads.
   - Add edges from each advertiser to t. Let the capacity of these edges be r_i, as we only care about the lower bound of the number of users to show ads.

add graph picture

§1.1.2 Algorithm and complexity

Complexity is something

Algorithm 1: Satisfy advertisers

```
// θ_exp is globally given, and initially set to ∞.
// θ_exp is globally given, and initially set to ∞.
// θ_exp is globally given, and initially set to ∞.
function ConstructGraph(DG, X, U, r, m, n)
    Let V be a set of vertices, E be a set of edges
    Add s and t to V
    // Add users
    for j from 1 to n:
        Add u_j to V
        Add edge (s, u_j) to E of capacity 1
    // Add demographic group
    for i from 1 to |DG|:
        Add dg_i to V
    for j from 1 to |U|:
        for dg in U_j:
            Add edge (u_j, dg) to E of capacity 1
    // Add advertisers
    for i from 1 to m:
        Add a_i to V
        for dg in X_i:
            Add edge (dg, a_i) to E of capacity r_i
        Add edge (a_i, t) to E from of capacity r_i
    return G = (V, E)
// The main function
function DynPCA(DG, X, U, r, m, n)
    G ← ConstructGraph(DG, X, U, r, m, n)
    Run the Ford Fulkerson algorithm on the network flow graph G
    return False
```

§1.2 Present the analysis on the correctness of your algorithm.

§1.2.1 Show that a solution to the original problem will result in flows (i.e. satisfies the conservation and capacity conditions) in the network flow graph G.

Proof. Suppose there exists an assignment of ads to users such that all advertisers are satisfied. Consider the solution where each advertiser a_i, will have r_i ads shown to one demographic group, $dg_k \in X_i$. We will show that we can obtain an $s - t$ flow from such assignment.

Let there be r_i number of user nodes, u_j, that belong to demographic group dg_k for each advertiser a_i. For each u_j and its advertiser a_i, consider the flow that sends one unit along each path of $s \to u_j \to dg_k \to a_i \to t$. Let $f(e) = 1$ for each $s \to u_j$ and $u_j \to dg_k$ edge. By definition, conservation condition holds for all u_j. The capacity condition holds for all $s \to u_j$ and $u_j \to dg_k$ edges because the capacities of those edges is 1.

We know that $f^{in}(dg_k)$ will be equal to the sum of r_i where advertiser a_i's demographic group was chosen to be dg_k. Let $f(e) = r_i$ for each $dg_k \to a_i$. Hence, conservation condition holds for all dg_k. The capacity condition holds for all $dg_k \to a_i$ because because the capacities of those edges is r_i.

Let $f(e) = r_i$ for each $a_i \to t$. Hence, conservation condition holds for all a_i. The capacity condition holds for all $a_i \to t$ because because the capacities of those edges is r_i.

Therefore, there exists an assignment of ads to users such that all advertisers are satisfied which results in a flow network. If all advertisers are satisfied, the total $s \to t$ flow will be $\sum_{i=1}^{m} r_i$. Otherwise, it will be less than the previous sum with flows less than r_i on edges $a_i \to t$. □

§1.2.2 Show that the solution to the network flow problem will give the solution for the original problem.

Proof. Suppose we have a solution with the $s \to t$ flow being $\sum_{i=1}^{m} r_i$. This means that $f(e) = r_i$ for at all edges $a_i \to t$ as the this is the only way to get total flow of $\sum_{i=1}^{m} r_i$. Consequently, this means we have identified r_i users for each a_i advertiser. Thus, all advertisers will be satisfied.

If the algorithm gives us a maximum $s \to t$ flow that is less than $\sum_{i=1}^{m} r_i$, it means $f(e) < r_i$ for at least one edge $a_i \to t$. Thus, some advertiser will not have the number of ads shown to users they requested. That is why in this case we can claim that there doesn't exist enough users such that all advertisers are satisfied. □