CISC621 Algorithm Design and Analysis, Spring 2016
Homework set V, problems 14, 15, 16,17, due Friday, May, 4pm

Check the "homework sheet" from the syllabus for general homework details. In particular, each homework solution is on an entirely separate (set of) sheet(s) of paper and is identified with your name(s). Do not staple solutions to two or more problems together.

14. [Individual Problem — island evacuation]

    The Delmarva peninsula is a water surrounded body of land including parts of the states of Delaware, Maryland, and Virginia. It is technically an island, having the Delaware Bay and Atlantic Ociean on the east, the Chesapeake Bay on the south and west and the C&D (Chesapeake and Delaware) Canal on the north. There are only a few bridges (6 or 7) from this island to the mainland. Similarly large, well populated islands include Manhattan and Staten Island in New York.

    In preparation for the need to evacuate the island in advance of a large storm such as a Hurricane, you are charged with calculating how rapidly the island can be evacuated. To simplify the model we will assume each person has a designated car they will use and that the cars are in designated locations. The function to be computed is EvTime(G,w,c,O) which returns the number of hours needed to evacuate the island. The input information is a graph, G = (V,E), depicting the road network along with a weight function, w(u,v), which records the capacity, in cars per hour, of the road segment from intersection u to intersection v. The c is a function on V we'll write as v.c for v in V. This v.c is the number of cars located at intersection v. For example a parking garage may be modeled as a vertex, g, with a single edge connecting to an adjacent road (the garage entrance), the road/entrance vertex being r. Then w(g,r) is the max rate that cars can flow out the entrance to the road, and g.c is the number of cars in the garage. Finally O is a set of vertices representing "off the island". Each vertex x in O has a single neighbor y not in O, the edge (y,x) being a bridge (or tunnel or ferry) from island to mainland. A car is considered evacuated when it arrives at a vertex in O.

    Design and analyze an efficient algorithm for determining how long it will take, given G,w,c,O, to evacuate the island so described, moving all cars at vertices in $V \setminus O$ to vertices in O without exceeding any of the road segment capacities designated by w.

    > Solution: Let s be a new vertex and connect it to each garage with weight w(s,g) = number of cars at g. Let t be a new vertex and connect it to each vertex b in O, with w(b,t) = infinity (or any number greater than the total number of cars). We want a flow from s to t that includes all the cars. Consider the flow possible in k hours: set the capacity of each edge (other than those incident on s) (u,v) to w'(u,v) = k*w(u,v), but w'(s,g) = w(s,g) remains the number of cars at garage g. Then if the max flow on this modified graph is the total number of cars (the cut separating s from all other nodes being the min cut), we see that the cars can flow off the island in k hours. To find the minimal value of k, use repeated doubling until k is sufficiently large but k/2 is not. Then use bisection on the interval k/2..k. The cost of this algorithm is O(lg(n)$\times$ (cost of one max flow computation)).

15. [Individual Problem — Space efficient Floyd-Warshall]

    Let G be a weighted graph given by weight matrix W. G may have negative weights but no negative cycles (including no negative self loops, no negative $w_{i,i}$). Show that the following version of Floyd-Warshall is correct. It uses $\Theta(n^2)$ memory rather than $\Theta(n^3)$. This is similar to exercise 25.2-4.

    FLOYD-WARSHALL"(W)

    1 n = W.rows.

    2 D = W

    3 for k = 1 to n

    4   for i = 1 to n if ($i \neq k$) then

```
        for j = 1 to n if (j ≠ k) then
            d_{i,j} = min(d_{i,j}, d_{i,k} + d_{k,j})
Return D.
```

Solution: For each k, we see that $d_{i,j}$ depends on it's value from smaller k and on the k-th row and col. Also the k-th row and col are not updated, but updating them would not change them (clearly $d_{i,k} = \min(d_{i,k}, d_{i,k} + \alpha)$ for any nonnegative α.) Thus if at the beginning of the k-th iteration $d = d^{k-1}$ then at the end of the k-th iteration $d = d^k$.

16. [Group Problem — fat graphs] Let V' be a subset of the vertices V of a graph G = (V,E). The subgraph *induced* by V' is the graph G' = (V',E'), where E' is the subset of E consisting of all edges with both ends in V'. For instance a k-*clique* of G is a subgraph induced by k vertices that turns out to be a complete graph and an *independent set* of vertices has no edges in its induced subgraph. Considering graphs to have many edges to be "fat" and graphs with few edges "thin", we could say that a clique is fat and an independent set is thin. To quantify the degree of fatness, call an induced subgraph (k,l)-fat if it has at most k vertices and at least l edges. The problem FAT(G,k,l) is to decide, given G,k,l, whether G has a (k,l)-fat subgraph. Show that FAT is NP Complete. Make clear your argument for each part of the recipe of NP Completeness proofs.

    Solution:

    (a) FAT(G,k,l) is in NP because a certificate can be a set of k vertices. The verifier counts the edges among these k vertices to see if it is at least l. The verifier can work in O($|E|$) time, checking each edge if it has ends in the set of k vertices. The certificate size is O(V).

    (b) Clique(G,k) reduces to FAT(G,k.l).
    The mapping can be to keep the same graph and k. Just set $l = k(k-1)$, then number of edges in a k-clique. Then if G has a k clique it has a $(k, k(k-1))$-fat subgraph. Conversely, if G has a $(k, k(k-1))$-fat subgraph, that subgraph is a k-clique. Thus Clique(G,k) if and only if FAT(G,k,k(k-1)).

17. [Group Problem — very independent] A *very independent* set F of G = (V,E) is a subset of V such that the neighborhoods of the nodes in F are pairwise disjoint. Thus for u and v in F, not only is (u,v) not an edge, but for every vertex w, at least one of (u,w) and (v,w) is not an edge also. Problem VIS (Very-Independent-Set) is: Given undirected graph G = (V,E) and integer k, decide if there exists in G a very independent set of size k

    Show that VIS is NP Complete.

    Solution:

    (a) Show VIS is in NP: Certificate can be a set of k vertices, Verification to show they are very intependent. Details skipped here.

    (b) Show IndependentSet reduces to VIS. It is not hard to show that vertices v,w are very independent if an only if there is no path of length 2 or less between them.
    The idea of the mapping is to put a vertex in the middle of each edge and then to connect all these middle-of-edge vertices. Another detail is needed so that we can relate very independent sets in G' to independent sets in G: that all these middle-of-edge vertices are connected to one more new vertex s which is the lone neigbor of an additional new vertex t..Note that now two of middle-of-edge vertices, s, and t are very independent.
    The mapping can be, given G=(V,E),k, to build a graph G' that has V' = V+E vertices, Label an vertex of G' as v if it is a vertex of G and label it vw if it is the middle of an edge (v,w) of G. The edges E' of G' are of three forms, the form (u, uv) or (uv, wx) or (uv, s). That is,

they connect an original vertex of G with a middle-of-edge vertex of G' if and only if the vertex is one end of the edge. The second form of edge in E' is every pair of middle-of-edge vertices (uv, wx). The third form connects every uv to s. Thus there are $|E'| = 3|E| + |E|^2$ edges in G'. If G has an independent set of vertices of size k, then that same set, with s adjoined, is very independent in G', since any path between them must visit at least 3 other vertices of G' including 2 middle-of-edge vertices. Now suppose G' has a set of k+1 very independent vertices. Since all the middle-of-edge vertices are connected, the set must contain at most one of those and if it does it does not contain s. We may replace the middle-of-edge vertex with s. Thus we if we have a k+1 very independent set we have one containing t and no middle-of-edge and no s. Suppose u and v are in the set and neither is t. The edge (u,v) must not be and edge of G, otherise G' would have the path (u, uv, v) of length 2, contradicting that u and v are very independent. We have shown that IndepSet(G,k) if and only VIS(G',k+1).