CS 38 Introduction to Algorithms	**Spring 2014**

Solution Set 4

Posted: May 13

If you have not yet turned in the Problem Set, you should not consult these solutions.

1. Denote by $x[i, j]$ to be the substring of x from character i to character j. If we define $\text{OPT}(i, j)$ to be the value of the highest quality decompostion of $x[i, j]$, then we can observe that $\text{OPT}(i, i) = q(x[i, i])$. If the string is not broken anywhere, then $\text{OPT}(i, j) = q(x[i, j])$; otherwise, an optimal decomposition breaks the string at some point and then comprises an optimal decomposition of the left and right substring, in which case

$$\text{OPT}(i, j) = \min_{k=i}^{j-1} \text{OPT}(i, k) + \text{OPT}(k + 1, j).$$

   Now we fill in the table by increasing order of $j - i$. There are $O(n^2)$ cells to fill in and each one requires running through $O(n)$ possible breakpoints. The overall running time is $O(n^3)$. To actually reconstruct the decomposition, as is standard, we record the optimal choice in each cell – either to not split or to split at position k. Then be working backwards from $\text{OPT}(1, n)$ we can output the optimal decomposition, in $O(n)$ additional time.

2. As in Bellman-Ford, we denote by $OPT(v, i)$ the optimal v-t using i or fewer edges. Clearly $OPT(v, 1) = \infty$ if there is no edge (v, t) and $OPT(v, 1) = c(v, t)$ if there is an edge (v, t). Then, as in class, either $OPT(v, i) = OPT(v, i - 1)$ if a (simple) shortest path uses only $i - 1$ edges, or $OPT(v, i) = OPT(w, i - 1) + c(v, w)$ for some edge (v, w) if the first edge of a (simple) shortest path is (v, w).

   Because the edge weights are all *positive*, every shortest s-t path is simple. Thus, we can *count* the number of shortest paths by storing a value in each cell with this count. More specifically, we store a count value in cell (v, i) which represents the *number* of (simple) shortest paths from v to t using i or fewer edges. This count is 0 for cell $(v, 1)$ if there is no edge (v, t) and 1 if there is an edge (v, t).

   To compute the count for cell (v, i), let $a = OPT(v, i-1)$ and $d(w) = OPT(w, i-1) + c(v, w)$; note that these are the length of a shortest path from v using at most $i - 1$ edges and the length of a shortest path starting with edge (v, w) and using at most $i - 1$ edges from w. Note that

$$OPT(v, i) = \min\{a, \min_{w:\text{edge } (v,w) \text{ exists}} d(w)\}.$$

   The count for cell (v, i) is thus the sum of the counts associated with the quantities that achieve the minimum; e.g. if the minimum is only achieved by a then we have only the count from cell $(v, i - 1)$; if the minimum is achieved by a and some $d(w)$ then we sum the count from cell $(v, i - 1)$ and the count from cell $v(w, i - 1)$.

In the end, the count in cell $(s, n-1)$ contains the number of (simple) shortest paths from s to t.

As in Bellman-Ford, the time to fill in a row of the table is $O(m)$ and there are n rows to fill in. Keeping track of the counts only multiplies this by a constant. The overall running time is thus $O(nm)$ as it was for the original Bellman-Ford algorithm.

3. Let $G = (U \cup V, E)$ be a bipartite graph. Define the matroid $\mathcal{I}_1$ to be those subsets of edges that contain no two edges incident to a vertex in U. Similarly define the matroid $\mathcal{I}_2$ to be those subsets of edges that contain no two edges incident to a vertex in V. Clearly, an independent set in the intersection of these two matroids is a subset of edges that contains no two edges incident to any vertex at all; i.e., a matching. Moreover a maximum cardinality independent set is a maximum matching in G.

   It remains to show that $\mathcal{I}_1$ is a matroid (and then the same holds for $\mathcal{I}_2$). Clearly $\mathcal{I}_1$ is closed under taking subsets, and it contains the empty set. Now, let A, B be two independent sets with $|A| < |B|$. Note that any independent set has either 0 or 1 edge incident to each vertex in U. Let S be the subset of vertices in U having one incident edge in A, and let T be the subset of vertices in U having one incident edge in B. Since B is larger, we have $|T| > |S|$ and there must be some edge incident to a vertex $v \in U \setminus S$. Adding this edge to A preserves independence. Thus all three axioms hold and we conlude that $\mathcal{I}_1$ is a matroid. The same argument shows that $\mathcal{I}_2$ is a matroid.

4. (a) Consider the bipartite graph $G = (U \cup V, E)$ and perform the reduction to maximum flow. We observe that we can send $1/k$ units of flow along every one of the edges of the original graph. Flow is conserved on the left because there is one unit of incoming flow and $(1/k) \cdot k = 1$ unit of outgoing flow; similarly flow is conserved on the right because there is $(1/k) \cdot k = 1$ unit of incoming flow and 1 unit of outgoing flow. The overall flow value is $|U| = |V|$ and thus there is a perfect matching by a Theorem from class.

   (b) In the forward direction, if there is a perfect matching, it is clear that by following just the edges in the perfect matching we find at least $|S|$ neighbors of the vertices in subset $S \subseteq U$, for every such subset as required.

   In the reverse direction, suppose there is *no* perfect matching. Then in the graph G' produced by the reduction to flow (with infinite capacities on the middle edges, as we noted was possible), we have a maximum flow value of strictly less than $|U|$. Consider the min-cut associated with this max-flow; i.e., the cut defined by those nodes reachable from the source in the residual graph. Notice that none of the middle edges can cross the cut as otherwise it would have infinite capacity, but the flow is not infinite. Thus the edges crossing the cut are either (1) edges from the source s to vertices in U, or (2) edges from vertices in V to the sink t. Suppose there are a of the former type of edges and b of the latter type. Let S be the subset of vertices of U contained in the s-side of the cut, and notice that (because not infinite capacity edges cross the cut) the vertices of V contained in the s-side of the cut are exactly $N(S)$. But we have $|S| = |U| - a$ and $|N(S)| = b$ and $a + b < |U|$, so

   $$|N(S)| = b < |U| - a = |S|,$$

   so there is a set for which $|N(S)| < |S|$ in case there is no perfect matching, as desired.

5. (a) We create a source s and a sink t, and we add directed edges from s to every node v in the original graph which has $d(v) < 0$ with capacity $-d(v)$ and we add directed edges from every node v in the original graph which has $d(v) > 0$ to the sink t with capacity $d(v)$.

   If a flow in this new graph saturates all of the new edges, we claim it yields a circulation in the original graph. Clearly the capacity constraints on the original edges are respected, and for each vertex with demand $d(v) > 0$, we have

$$\sum_{e \text{ entering } v} f(e) - \sum_{e \text{ exiting } v} f(e) = d(v),$$

   because the extra edge from v to t accounts for $d(v)$ units of exiting flow. Similarly, for each vertex with demand $d(v) < 0$, we have

$$\sum_{e \text{ entering } v} f(e) - \sum_{e \text{ exiting } v} f(e) = d(v),$$

   becaue the extra edge from s to v accounts for $-d(v)$ units of incoming flow. Thus we can read off the circulation by looking at the flow assigned to the edges of the original graph.

   We also need to prove that if there is a circulation, then there is a flow that saturates all the new edges. This is easy: simply assign flow to each edge in the original graph according to the circulation, and then there is precisely enough excess flow at vertices with $d(v) > 0$ to saturate the edge from v to t; and there is precisely enough missing flow at vertices with $d(v) < 0$ to saturate the edge from s to v.

   So there is a circulation iff the maximum flow saturates all of the new edges.

   (b) We modify the circulation graph and demands as follows: we replace each edge $e = (v, w)$ with three edges (v, v'), (v', w') and (w', w). We need to specify the demands for the two new vertices. We set $d(v') = -\ell(e)$ and $d(w') = \ell(e)$.

   Now, any circulation f in the original graph that satisfies the lower bounds can be extended to a circulation in this new graph by making the flow on edge (v, w') equal to $f(e)$, the flow on edge (v', w') equal to $f(e) - \ell(e)$ and the flow on edge (w', w) equal to $f(e)$.

   In the other direction, any flow on the new graph must satisfy $f(v, v') = f(v', w') - \ell(e)$ and $f(w', w) = f(v', w') + \ell(e)$. Thus the flow on edge (v, v') and the flow on edge (w', w) are equal, and we can set the flow on edge e in the original graph equal to this value to produce a circulation in the original graph (since all the demand-equations at original vertices remain satisfied by this choice). Moreover, since $f(v', w)$ must be nonnegative, we find that $f(v, v') \geq \ell(e)$ as desired.

   (c) We add new vertices s and t, and add edges from s to each $c \in C$, with capacity q'_c and lower bound $\ell(s, c) = q_c$. We add edges from each $p \in P$ to t with capacity s'_p and lower bound $\ell(p, t) = s_p$. Finally, we add an edge from t to s with capacity ∞ (and no lower bound; i.e. a lower bound of 0). The capacity on each edge of the original bipartite graph is set to 1 with a lower bound of 0. The demands at all vertices are set to 0.

   Now, if the specified set of survey parameters are feasible, then by setting $f(s, c)$ to be the number of questions asked consumer c, and $f(p, t)$ to be the number of questions asked about product p, we satisfy the flow constraints for these edges. Since each of the q questions asked to a single consumer is about a different product, we have capacity to route all the flow arriving at c to the q different vertices in P. Finally, all the flow arriving at t can be sent back to s via the infinite capacity edge to make this a valid circulation.

On the other hand, if a circulation exists, then we claim that the survey parameters are feasible. By the integrality theorem for flows (which extends to circulations and circulations with lower bounds since in each case we reduce to solving flow), all the internal edges can be assumed to have flow 0 or 1. The fact that the original edges of the bipartite graph have capacity 1 then means that if consumer c is asked q questions, they are about q different products. Moreover, since the circulation satisfies the lower bounds and capacity constraints, each consumer c is asked between q_c and q'_c questions, and each product is surveyed between s_p and s'_p times.