CS 38 Introduction to Algorithms	**Spring 2014**

Solution Set 2

Posted: April 22

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

1. (a) Assume without loss of generality that the $\ell_1 \leq \ell_2 \leq \cdots \leq \ell_n$. Consider the full binary tree of height $h = \ell_n$. Partition the 2^h leaf nodes (considered in the natural order from left to right) into blocks of size $2^{h-\ell_1}, 2^{h-\ell_2}, \ldots, 2^{h-\ell_n}$. The sum of these sizes is $2^h \cdot 2^{-(\ell_1+\ell_2+\cdots+\ell_n)} \leq 2^h$ by assumption. Now remove the subtree whose leaves constitute the first block, then the subtree whose nodes constitute the second block, and so on. The roots of theses subtrees have depths $\ell_1, \ell_2, \ldots, \ell_n$ as desired.

   (b) Set $r_i = \lfloor \log_2(1/p_i) \rfloor$, and note that $\sum_i 2^{-r_i} \leq \sum_i p_i = 1$ since $\mathbf{p}$ is a probability distribution. By part (a) there is a prefix-free encoding scheme with lengths r_i. Since the ℓ_i represent the lengths of an *optimal* prefix-free encoding scheme, we have

   $$\sum_i \ell_i p_i \leq \sum_i r_i p_i = \sum_i \lfloor \log_2(1/p_i) \rfloor p_i \leq \sum_i (\log_2(1/p_i) + 1) p_i = H(\mathbf{p}) + 1,$$

   as required.

2. (a) For both the graphic and matric matroid, the first two axioms clearly hold. We focus on the third property, sometimes called the exchange property.

   For the graphic matroid, consider two forests F_1 and F_2, with $|F_1| < |F_2|$. We claim there is an edge in F_2 between two distinct connected components of F_1. Suppose for the purpose of contradiction that each connected component of F_2 is contained in a connected component of F_1. Now F_1 has tree on each of its connected components, which has the maximum number of edges for an acyclic subgraph on that component. So F_2 has at most as many edges as F_1 in each connected component of F_1, which implies $|F_2| \leq |F_1|$, a contradiction. Thus we can add an edge of F_2 to F_1 while maintaining independence, as required.

   For the matric matroid, consider two subsets of columns, C_1 and C_2, with $k = |C_1| < |C_2|$. We must have $|C_1|$ strictly less than the column-rank of the matrix, since C_2 is larger and independent. Now if every column vector in C_2 is contained in the span of C_1, then C_2 would not be independent (we cannot have more than k independent vectors in the span of a k-dimensional subspace). Adding this vector to C_1 maintains independence, as required.

   (b) We sort the elements of the universe E in order of decreasing weight in $O(|E| \log |E|)$ time, and we begin adding the elements in this order, only skipping over an element when we find that adding it to the set so far yield a set that is not independent. This clearly has the required running time. Now consider the sequence of sets

   $$\emptyset = A_0 \subseteq A_1 \subseteq A_2 \subseteq \cdots \subseteq A_n = A$$

obtained as the algorithm considers the n elements of E in order of decreasing weight. We claim that for all i, A_i is contained in some maximum-weight independent set B_i for which $B_i \setminus A_i$ is contained in elements $i+1, i+2, \ldots, n$ (the elements not yet considered). This holds for A_0 by taking B_0 to be any maximum weight independent set.

Now suppose it holds for $i-1$, and consider A_i, which is formed from A_{i-1} by considering the i-th element. If the i-th element is not added, then $A_{i-1} \cup \{i\}$ is not independent, and then since $A_{i-1} \subseteq B_{i-1}$ we find that B_{i-1} cannot include i (if it did, then by axiom 2, we could remove elements of B_{i-1} until it comprised just $A_{i-1} \cup \{i\}$. So in this case we can take $B_i = B_{i-1}$. Otherwise, the i-th element is added. If B_{i-1} contains i, then we can take $B_i = B_{i-1}$. If not, then by repeated application of axiom 3 to A_i and basis B_{i-1}, we can extend A_i to a basis B using elements of B_{i-1}. This process adds all but one element of B_{i-1}, call it j, so we have $B = B_{i-1} - \{j\} + \{i\}$. Since $B_{i-1} \setminus A_{i-1}$ contains only elements from $\{i+1, \ldots, n\}$, the weight of element j is no larger than that that of element i. Thus the weight of B is at least as large as the weight of B_{i-1}, which was a maximum-weight independent set. We can then take $B_i = B$.

We conclude that $A_n \subseteq B_n$ where B_n is a maximum-weight independent set, and because A_n is as well, we have $A_n = B_n$, which completes the proof.

(c) The first two axioms are obvious. For the third axiom, consider two independent sets A and B in the dual, with $|A| < |B|$. Then there is a basis X of the original matroid that is disjoint from B. There is also a basis Y of the original matroid disjoint from A such that

$$X \setminus A \subseteq Y$$

since we can augment $X \setminus A$ with elements of a basis disjoint from A. We claim that $B \setminus A$ contains an element not in Y. In this case we can add this element to A and the resulting set is still disjoint from Y, and therefore is an independent set in the dual as required. To verify the claim, observe that if it fails, then we can compute $|X|$ as

$$|X \cap A| + |X \setminus A| \leq |A \setminus B| + |X \setminus A| < |B \setminus A| + |X \setminus A| \leq |Y|$$

where the first inequality holds because X and B are disjoint, and the second inequality holds because $|A| < |B|$, and the third inequality holds because we are assuming $B \setminus A$ is contained in Y and moreover B and X are disjoint. This contradicts that X and Y could both be basis (since they have different cardinalities).

(d) Clearly the first two axioms hold; for the 3rd, consider A, B both of cardinality at most k, for which $|A| < |B|$. Then the 3rd axiom holds for these sets, and so $\mathcal{I}_k$ is a matroid as required.

3. (a) We already argued that the graphic matroid is indeed a matroid. If w_e are the weights for each edge e, then we assign to element e in the matroid, the weight $L - w_e$, for L equal to the maximum weight among all edges. Since every basis has the same cardinality k, we have that the weight of any maximum-weight basis B, equals $kL - \sum_{e \in B} w_e$, which is maximized when B is a basis minimizing $\sum_{e \in B} w_e$, as desired.

(b) The first two axioms are trivial to verify. Consider two independent sets A, B with $|A| < |B|$. If A contains no cycle, then adding any edge of B created at most one cycle, so the axiom holds. If A contains a cycle with an edge e not in B, then we can remove edge e from A and an edge on a cycle in B (if there is one), resulting in A' and B', with $|A'| \leq |B'|$ and both A' and B' being forests. As we know from the fact that the graphic matroid is a matroid, we can add an edge of B' to A' without creating a cycle. Then adding e back creates at most one cycle, and we are done. The last remaining case is that A contains a cycle that is completely contained in B. In this case consider removing an edge e from the cycle from both A and B, and applying the analysis of the graphic matroid to the resulting sets A' and B'. We find that there is an edge of B' that can be added to A', that connects two distinct components. Since only one of these components could have contained the cycle, we can create only one cycle when adding back edge e. Thus the third axiom holds.

(c) Let $\mathcal{I}$ be the dual of the graphic matroid on the underlying graph G. The bases of this matroid contain exactly the maximum sets of edges that can be removed from G while leaving it connected (since they are complements of spanning trees). Then $\mathcal{I}_k$ has bases which are exactly the sets of k edges that can be removed without disconnecting G, as required.

4. We perform a DFS traversal of the tree, starting at the root. Each time we finish (not discover) a vertex v, we perform a union with its parent, and “name” the resulting set in the union-find data structure by the parent. This maintains the invariant that at the time subtree T with parent w is finished, all of its nodes are in a single set in the union-find data structure, named w.

   Consider two nodes (u, v) whose least common ancestor is node w, and suppose that u's subtree is explored first. Then at the point that we discover v, the set in the union-find data structure containing v is the one named by w. So, as we discover each vertex v we see if it belongs to any pairs in the input. If its “partner” u has been finished, we output the name of the set resulting from a find on u. As we have argued this will be the least common ancestor of the pair.

   To support this last part, we should store the pairs as an array of linked lists, with element u of the array containing a linked list of all nodes v for which the pair (u, v) is in the input list of pairs. We can easily produce this structure in linear time by scanning the list of pairs once. Then, observe that the algorithm only examines v's list once when it discovers v, so the overall time for these checks is linear in m.

   Finally, we perform n make-set operations (one for each vertex), and n union operations (each time we finish a vertex), and m find operations, once for each pair in the input list. The resulting cost using the union-find data structure is $O((m+n)\log^* n)$, and all other operations cost $O(m+n)$ in the aggregate.