Carsten Hood UIN: 922009787
CSCE 411-200 Fall 2016
Homework 5
Due Wed, Nov 9, 11:30 AM

— — — — — — — — — — — — — —

1. Exercise 24.3-8 (p. 664). Let G = (V, E) be a weighted, directed graph with nonnegative weight function w : E -> {0, 1, …, W} for some nonnegative integer W. Modify Dijkstra's algorithm to compute the shortest paths from a given source vertex s in O(WV + E) time.

Given these limited edge weights, we can adapt our priority queue implementation to achieve O(WV + E) time. Edges have at most a weight of W. Hence the longest possible distance between two vertices is (V-1)*W. Since all weights are whole numbers between 0 and W, all distances must be whole numbers between 0 and (V-1)W. Therefore we can implement the priority queue with an array of size (V-1)W+2, with boxes to represent infinity, zero, and the numbers 1 through (V-1)W. Each box in the array holds a linked list of vertices having identical distances corresponding to their index.

All inserts take a total of O(WV) time, as the array is initialized with (V-1)W+2 or O(WV) boxes and inserting all V items at the front of the linked lists (initially just the linked list associated with distance infinity) takes constant time per insert or O(V) total. All extract-min operations will move one direction through the array, taking O((V-1)W+2) or simply O(WV) time. Decrease-key operations are constant, as items can be removed from and inserted at the front of linked lists within the array at constant time; hence we have a total of O(E). Add these totals together: O(WV) + O(WV) + O(E) = O(WV + E).

— — — — — — — — — — — — — —

2. Exercise 5.2-5 (p. 122). Let A[1. . n] be an array of n distinct numbers. If i < j and A[i] > A[j] , then the pair (i, j) is called an inversion of A. (See Problem 2-4 for more on inversions.) Suppose that the elements of A form a uniform random permutation of (1, 2, …, n). Use indicator random variables to compute the expected number of inversions.

Let I_{ij} be the indicator random variable for the pair (i, j) constituting an inversion, where $i \leq j$ and both i and j are indices of the array. There are a total of $1 + 2 + \cdots + n - 1$ or $n(n - 1)/2$ total such pairs. Each pair's probability of being inverted, $E[I_{ij}]$, is $1/2$. The desired expected number of inversions is the number of pairs times each pair's probability of constituting an inversion:
$E[I_{ij}] * \frac{n(n-1)}{2} = \left(\frac{1}{2}\right) * \frac{n(n-1)}{2} = \frac{n(n-1)}{4}$.

— — — — — — — — — — — — — —

3. Exercise 7.4-4 (p. 184). Show that randomized quicksort's expected running time is $\Omega(n \lg n)$.

As explained in the notes, the running time of quicksort is proportional to the number of comparisons, and the expected total number of comparisons can be defined as: $\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1}$. We will manipulate this to show that it is bounded by $\Omega(n \lg n)$, substituting k for $j - i$ as in the notes:

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1} = \sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{2}{k+1} \geq \sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{2}{2k} = \sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{1}{k} = \sum_{i=1}^{n-1} \Omega(\lg n) = \Omega(n \lg n).$$

— — — — — — — — — — — — — —

4. Exercise 26.2-9 (p. 731). For each of the two properties, either prove that every such augmented flow satisfies the property or give a counter-example.
"Suppose that both *f* and *f'* are flows in a network G and we compute flow *f* |^ *f'*. Does the augmented flow satisfy the flow conservation property? Does it satisfy the capacity constraint?"

First we show that $f \uparrow f'$ satisfies flow conservation. Since f and f' are valid flows we know that their flow conservation properties hold, i.e., $\sum_{v \in V} f(v,u) = \sum_{v \in V} f(u,v)$, and likewise for f'. We must show the same for $f \uparrow f'$. Start with $\sum_{v \in V} f \uparrow f'(v,u)$:

$= \sum_{v \in V} f(v,u) + \sum_{v \in V} f'(v,u)$ (since $f \uparrow f'(v,u) = f(v,u) + f'(v,u)$)
$= \sum_{v \in V} f(u,v) + \sum_{v \in V} f'(u,v)$ (since f and f' necessarily satisfy flow conservation)
$= \sum_{v \in V} (f(u,v) + f'(u,v))$ (combining the sums)
$= \sum_{v \in V} f \uparrow f'(u,v)$ (which proves that flow conservation is satisfied).

The capacity constraint property is not necessarily satisfied since the combined flows could surpass some edge's capacity. As a counterexample, let edge (a,b) have capacity 10. Let $f(a,b) = 6$ and $f'(a,b) = 6$, each clearly satisfying (a,b)'s capacity. But then $f \uparrow f'(a,b) = 12 \geq 10$, which violates the capacity constraint.

— — — — — — — — — — — — — —

5. Exercise 26.3-2 (p. 735). "Prove Theorem 26.10 (Integrality theorem): If the capacity function *c* takes on only integral values, then the maximum flow *f* produced by the Ford-Fulkerson method has the property that |*f*| is an integer. Moreover, for all vertices *u* and *v*, the value of *f(u,v)* is an integer.

"The proof is by induction on the number of iterations."
The initial flow of all 0s of the first iteration of Ford-Fulkerson clearly satisfies the property of $|f|$ being an integer value. Subsequent flows result from adding a flow f_p along an augmenting path of the residual network, whose value equals the smallest capacity of any edge in the path. Since all edge capacities only take on integral values, the value of f_p must also be an integral value. Because summing two integral values produces an integral value, each subsequent iteration's flow value, including the resulting flow value, will be an integral value. By an analogous proof, because the flow across one edge starts at 0 and will only increase by its or other edges' capacities over the course of the algorithm, we know that $f(u,v)$ must also constitute an integral value.