Algorithms: Assignment sheet 1

Due date: October 30, 2020

---

1. We saw that $n-1$ comparisons are necessary and sufficient to find the minimum element in an unsorted array of n elements. Show a comparison based algorithm for finding the minimum *and* maximum in an unsorted array of n elements using $\lceil 3n/2 \rceil - 2$ comparisons. Also show that $\lceil 3n/2 \rceil - 2$ comparisons are necessary to find the minimum and maximum.

2. Show that the minimum and second minimum elements in an unsorted array of n elements can be determined using $n + \log n - 2$ comparisons. Assume n to be a power of 2.

3. Consider two sets A and B, each having n integers in the range from 0 to $10n$. We wish to compute the *Cartesian sum* of A and B, defined by

   $$C = \{x + y : x \in A, y \in B\}.$$

   Note that the integers in C are in the range 0 to $20n$. We want to find the elements in C and the number of times each element of C is realized as a sum of elements in A and B. Show that the problem can be solved in $O(n \log n)$ time.

4. We are given 2 positive integers $\langle a_{n-1} a_{n-2} \dots a_0 \rangle$ and $\langle b_{n-1} b_{n-2} \dots b_0 \rangle$, of n bits each, as 2 arrays $A[1..n]$ and $B[1..n]$, respectively. The MSB of a, that is a_{n-1}, is stored in the location $A[1]$ and so on. Similarly with b. Give an $O(n \log n)$ algorithm to compute the array $C[1..2n]$ that holds the integer ab (the product of a and b).

5. Show that the number of distinct minimum cuts in an undirected graph (assume all edge weights are one) is at most $\binom{n}{2}$. That is, show that the number of distinct cuts whose value is equal to the value of the min cut in the graph is at most $\frac{n(n-1)}{2}$.

6. Suppose that at each step of the simple contraction algorithm for min-cut, instead of choosing a random edge for contraction, we choose two vertices at random and merge the two vertices into a single vertex. Show that there are inputs on which the probability that this modified algorithm finds a min-cut is exponentially small.

7. Show the following flow decomposition theorem:

   Any s-t flow f can be decomposed into at most m "primitive elements", where a primitive element is:

   (i) a path P from s to t where the flow on each edge of the path is $\delta(P)$, or

   (ii) a simple cycle Γ where the flow on each edge of the cycle is $\delta(\Gamma)$.

   [Note that the decomposition is not unique. The various paths and cycles are not necessarily edge disjoint - i.e., an edge can be a member of multiple primitive elements. The flow on an edge includes the flow of all paths and cycles that include the edge.]

8. Let $G = (V, E)$ be a flow network with source s, sink t, and integer capacities. Let $|E| = m$ and $|V| = n$. Assume that we are given a maximum flow in G.

   (a) Suppose we increase the capacity of a single edge $(u, v) \in E$ by 1. Give an $O(m + n)$ algorithm to update the maximum flow.

   (b) Suppose we decrease the capacity of a single edge $(u, v) \in E$ by 1. Give an $O(m + n)$ algorithm to update the maximum flow.

9. Let G be an undirected graph and let x and y be any two vertices in G. Show that x-y minimum cut (i.e., the minimum number of edges whose removal disconnects x and y) is equal to the maximum number of edge disjoint paths between x and y in G.

10. Consider the following network - the numbers next to the edges denote their capacities and the capacity r of the edge (a, b) is $(\sqrt{5} - 1)/2$.

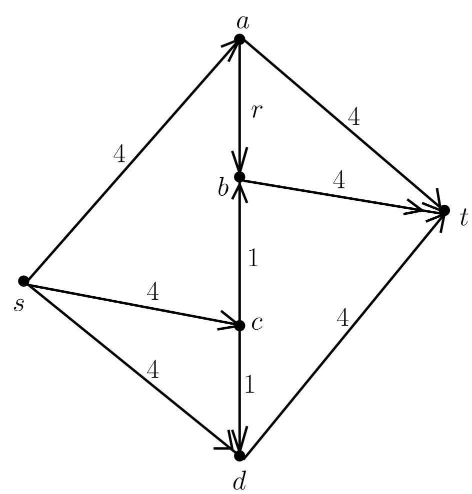

Figure 1: An example where Ford-Fulkerson algorithm does not terminate.

Let the initial flow be along the path $p_0 = (s, c, b, t)$. Take $p_1 = (s, a, b, c, d, t)$, $p_2 = (s, c, b, a, t)$, $p_3 = (s, d, c, b, t)$.

Show that $p_0(p_1, p_2, p_1, p_3)^*$ is an infinite sequence of s-t paths along which positive flow can be sent. The flow value does not converge to the maximum value 9.