CS5800 Algorithms

Recitations from Dec 2, 2019 to Dec 6, 2019

Ravi Sundaram

Problem 1 (Review)

1. You are given an $n \times n$ matrix $A[1...n][1...n]$ of integers. An element in the matrix is called a peak if its greater than or equal to its four neighbors (left, right, above, below).

   If an element is on the edge of the matrix, then missing neighbors are considered to have the value $-\infty$. largest For example, in the following matrix

   $$A = \begin{bmatrix} 5 & 14 & 10 \\ 16 & 20 & 8 \\ 4 & 6 & 12 \end{bmatrix}$$

   20 and 12 are both peaks.

   Give an $O(n \log n)$-time algorithm to find a peak in a given $n \times n$ matrix.

2. You are given a sequence $A = (a_1, a_2, \ldots, a_m)$ and you need to find the length of the longest palindromic subsequence of A. A subsequence is palindromic if the reverse of the subsequence is equal to the subsequence itself.

   For example in $A = (I, N, T, E, R, N, E, T)$, then the longest palindromic subsequencec is $A' = (T, E, R, E, T)$ and the length is 5.

   (a) If you had black-box access to an algorithm that outputted the Longest Common Subsequence (LCS) of two sequences A and B, show how to use that to solve the Longest Palindromic Subsequence (LPS) problem.

   (b) Assuming you did not have access to LCS, solve this using dynamic programming. You only need to write the English description of your DP and the Bellman Equation (including base cases).

3. Formulate Clique as an integer linear programming problem. Recall,

   Definition (Clique)**.** *A* clique *in a graph $G = (V, E)$ is a subset $V' \subseteq V$ such that, for every pair of vertices $u, v \in V'$, the edge $\{u, v\} \in E$.*

   Problem (Clique)**.** *Given a graph $G = (V, E)$ and a positive integer $k \leq |V|$, does G contain a clique V' having $|V'| \geq k$?*

4. Consider the following LP:

$$\begin{aligned} \min \quad & 3x + 8y + 4z \\ \text{s. t.} \quad & x + y \geq 8 \\ & 2x - 3y \leq 0 \\ & y \geq 9 \\ & x, y \geq 0 \end{aligned}$$

   (a) Write the LP in Standard Form.

   (b) Write the LP in Canonical Form.

   (c) Write the dual of the LP.

   Note: If you need to use slack variables, call them $s_1, s_2, \ldots$ and if you need to enforce a non-negativity condition on an unconstrained variable a use the a^+ and a^- superscript notation.

Solution:

1. Our divide and conquer algorithm is as follows. We start at the middle column (at index $m = \lfloor \frac{1+n}{2} \rfloor$). We scan all the entries in column m (so the elements $A[1][m], A[2][m], \ldots, A[n][m]$) and find the entry with the maximum element. Let us denote this entry by $A[k][m]$. We know that $A[k-1][m] \leq A[k][m] \geq A[k+1][m]$. So then we check if $A[k][m-1] \leq A[k][m] \geq A[k][m+1]$. If so, then our algorithm can terminate because we just found a peak. Otherwise, either $A[k-1][m] \geq A[k][m]$ or $A[k+1][m] \geq A[k][m]$. Suppose $A[k-1][m] \geq A[k][m]$. Then we know there will be a peak in the left half of the array, so we recurse to that side. We then take the middle column $m' = \lfloor \frac{1+m}{2} \rfloor$, and repeat what we just did. If $A[k+1][m] \geq A[k][m]$, then we recurse on the right side.

   We are recursing on the number of columns. We start with n columns and eliminate half of them each iteration, so there are $O(\log n)$ iterations. But at each iteration, we scan a column for the maximum element, which takes $O(n)$ time, so the total runtime is $O(n \log n)$.

2. (a) We can take our input sequence A, create a copy of it and then reverse it. Let's call this reversed sequence A'. We then find the LCS of A and A', and that will be the longest palindromic subsequence of A.

   (b) English Description of recursive subproblem: We create a new array PAL of dimension $n \times n$, where $PAL[i, j]$ will contain the length of the longest palindrome subsequence in the subarray $A[i, i+1, \ldots, j]$.

   English Description of logic: It's easy to see that if $A[i] = A[j]$, for some $i < j$, we can include that in palindrome subsequence and increase i by 1 and decrease j by 1. And if they're not equal, we either skip the ith entry, or the jth entry, and take the maximum of the two longest palindromic subsequences in those two subarrays.

$$PAL[i,j] = \begin{cases} PAL[i,j] = 0 & i > j \\ PAL[i,j] = 1 & i = j \\ max(PAL[i, j+1], PAL[i-1, j]) & A[i] \neq A[j] \\ 2 + PAL[i-1, j+1] & A[i] = A[j] \end{cases}$$

3. So the definition of a clique S means that for every pair of vertices $u, v \in S$, there is an edge between them, $(u, v) \in E$. We need to come up with some variables for our linear program. A reasonable idea is to create a variable $x_v \in \{0, 1\}$ for each vertex v, where $x_v = 1$ represents v being in the clique, and $x_v = 0$ represents v not being in the clique.

   If we have two vertices $x_u = x_v = 1$, then we need to somehow make it required that there's an edge between through a linear inequality. To do this, we introduce the variables e_{uv} for each pair of vertices $u, v \in V$. These are simply indicator variables that indicate if the edge $(u, v) \in E$. If $(u, v) \in E$ then $e_{uv} = 1$, otherwise $e_{uv} = 0$.

   Let's take the sum $x_u + x_v$. If it's 0 or 1, then we don't care what e_{uv} is. However if its 2, then e_{uv} has to be 1. Now let's take the expression $x_u + x_v - e_{uv}$. Note that the maximal value this expression can take is 2, and that's only in the case if $x_u + x_v = 2$ and $e_{uv} = 0$, which is precisely the case we need to make illegal. So we can lower bound the inequality by 1.

   The linear program below will find the clique of maximal size in the given graph $G = (V, E)$.

$$\begin{aligned} \max \quad & \sum_{v \in V} x_v \\ \text{subject to} \quad & x_u + x_v - e_{uv} \leq 1 \quad \forall u, v \in V \\ & x_u \in \{0, 1\} \quad \forall u \in V \\ & e_{uv} = 1 \quad \forall (u, v) \in E \\ & e_{uv} = 0 \quad \forall (u, v) \notin E \end{aligned}$$

4. (a)

$$\begin{aligned} \max \quad & -3x - 8y - 4z^+ + 4z^- \\ \text{s. t.} \quad & x + y - s_1 = 8 \\ & 2x - 3y + s_2 = 0 \\ & y - s_3 = 9 \\ & x, y, z^+, z^-, s_1, s_2, s_3 \geq 0 \end{aligned}$$

   (b)

$$\begin{aligned} \min \quad & 3x + 8y + 4z^+ - 4z^- \\ \text{s. t.} \quad & x + y \geq 8 \\ & -2x + 3y \geq 0 \\ & y \geq 9 \\ & x, y, z^+, z^- \geq 0 \end{aligned}$$

   (c) Formally we get

$$\begin{aligned} \max \quad & 8a + 9c \\ \text{s. t.} \quad & a - 2b \leq 3 \\ & a + 3b + c \leq 8 \\ & 0 \leq 4 \\ & 0 \leq -4 \end{aligned}$$

But this linear program obviously has no feasible solution because of the contradiction in the constraints. Note that the original linear program is unbounded. In this case, the dual program won't have a feasible solution. This makes sense through the duality theorems of linear programs.

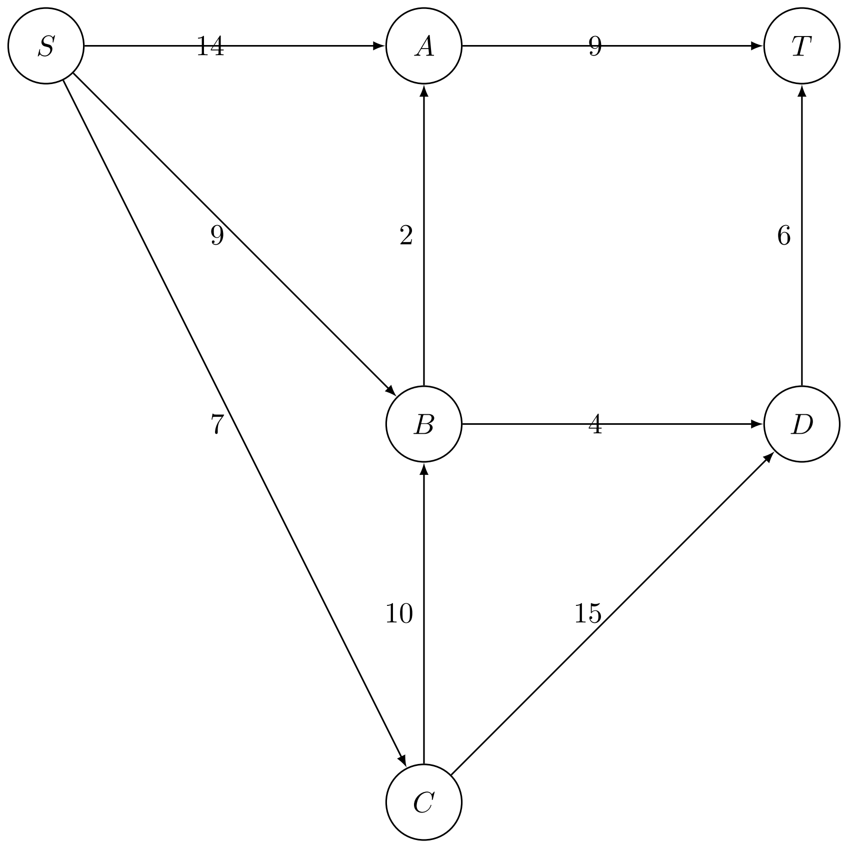
□

Problem 2 (More Review)

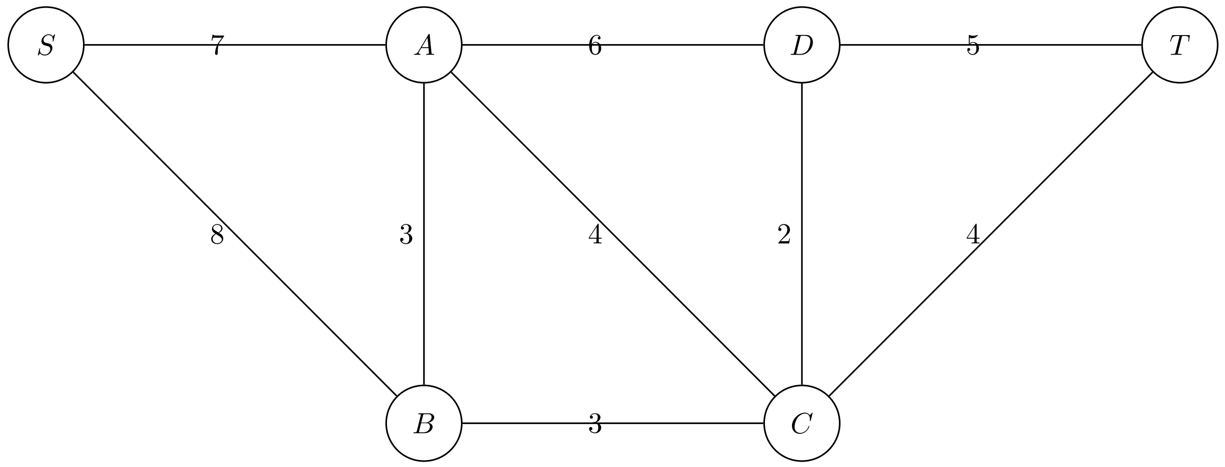
1. For some given functions, f, g, h, decide whether each of the following statements is true or false.
   (a) If $f(n) = O(g(n)), h(n) = O(g(n))$, then $f(n) = \Theta(h(n))$
   (b) If $f(n) = \Omega(g(n))$, $g(n) = \Theta(h(n))$, then $f(n) = \Omega(h(n))$
2. Solve the following recurrences. First, solve it by drawing out the recurrence tree. Next, verify your answer by the Master's Theorem.

$$T(n) = 10T(n/3) + n^2$$
$$T(n) = 8T(n/3) + n^2$$
$$T(n) = 9T(n/3) + n^2$$

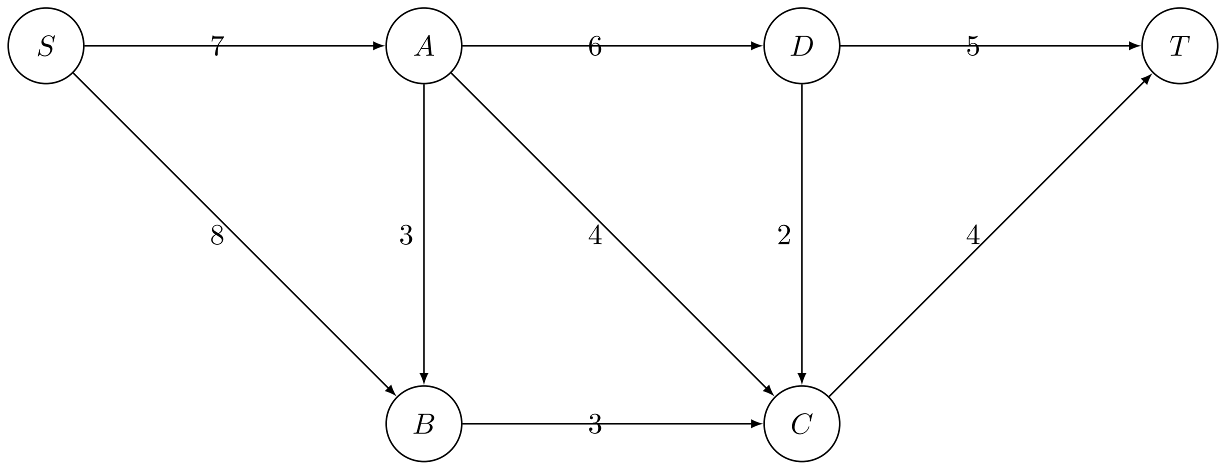
3. Say you are given the graph $G = (V, E)$ where $V = \{a, b, c, d, e, f, g\}$, $E = \{(a, b), (b, d), (d, f), (a, e), (c, d)(a, c), (c, e), (b, e), (e, g), (d, g), (f, g)\}$. Do a BFS traversal on this graph. Next, do a DFS traversal. We put vertices in the queue/stack based on their order in the alphabet.
4. Execute Dijkstra's to find the shortest path from s to every other vertex on the following graph.



5. Execute Kruskal's algorithm to find the minimum spanning tree on the following graph. Then execute Prim's algorithm.



6. Execute Edmonds-Karp to find the max flow on the following graph. Write the augmenting path at each iteration and how much flow goes through the augmenting path.



7. You are a chef in a kitchen. You are in possession of n different kinds of ingredients, with u_i units of ingredient i. You know how to make m different kinds of dishes, and making one dish d_j requires exactly one unit of exactly one ingredient u_{d_j}. One dish feeds exactly one customer. A dish can be made more than once and that'll feed more than one customer. You have p customers waiting to eat. Due to dietary restrictions, a customer c_k can only eat from a subset of the dishes $D_k \subseteq \{d_1, d_2, ..., d_m\}$. Determine whether you are able to feed every customer a dish.

8. Show that CLIQUE is NP-Hard by reducing from 3SAT.