1 Recursion Trees (10 Points)

Use a recursion tree to generate a guess of the asymptotic complexity of a divide and conquer algorithm with the running time $T(n) = 2T(n/3) + O(n^2)$.

2 Master Theorem (9 Points)

Solve the following recurrences using the master theorem. Justify your answers shortly (i.e. specify ϵ and check the regularity condition if necessary).

1. $T(n) = 3T(n/4) + n^0$

2. $T(n) = 4T(n/2) + n^0$

3. $T(n) = 3T(n/2) + n^2 \log n$

3 Greedy Algorithms (14 Points)

Suppose we are driving an electric vehicle across the country on an interstate highway that has charging stations set up along the interstate. Suppose we have an array L of locations where $L[i]$ is the location of the ith charging station in terms of miles from the start of our drive. We can assume that L is sorted in increasing order (the order of L is the order in which we will encounter them on our drive). Suppose that our vehicle cannot go more than 250 miles without a charge, and subject to this, we want to stop to charge our vehicle as few times as possible. Give a greedy algorithm to compute the minimum number of stops we need to make, and argue why your algorithm is correct.

4 Dynamic Programming (16 Points)

Suppose we work for a company that makes electric vehicles, and we are planning to install charging stations along an interstate highway that extends across the country. Further suppose we asked n businesses along the interstate for proposed costs for us to be able to install a charging station on their property. If a business tells us they will let us install for cost c, then we will either pay them c to install at their location, or we will not install at their location and pay 0. We want to ensure that we have a charging station at least every 100 miles, and subject to this constraint, we want to minimize the sum of the costs of the charging stations. Suppose C is an array containing the costs ($C[i]$ is the cost for business i), and L is an array containing the locations ($L[i]$ is how many miles the ith business is from the beginning of the interstate). You can suppose that for any 100 mile stretch of the interstate, there is at least one business (otherwise a feasible solution does not exist).

1. A potential greedy algorithm might be to repeatedly take the cheapest location until we have a feasible solution (no 100 mile stretch without a charging station). Give a counterexample to show that this algorithm will not always produce an optimal solution.

2. Let $A[i]$ denote the cost of an optimal solution that only considers the first i businesses (i.e., we suppose the road ends at $L[i]$ and we ignore all business j such that $j > i$). Consider the following example. Suppose the interstate is 300 miles long. What is $A[1], A[2], A[3], A[4]$, and $A[5]$ for this example?

   C = [13, 14, 11, 16, 11].

   L = [60, 105, 148, 223, 285].

3. Give a recursive definition for $A[i]$. Do not forget the base case. Note you don't have to give a full algorithm here. Just define A[i] in terms of its subproblems.

5 Graph Algorithms (15 Points)

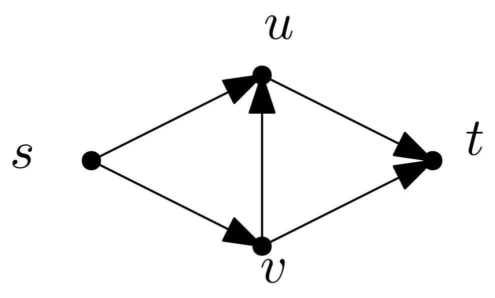
Recall that a clique is a subset C of vertices of a graph such that every pair of vertices in C has an edge connecting them. So for example, if the graph is the Facebook graph, a clique C would represent a subset of people who are all friends with each other. In general, determining if a graph G has a clique of size k is NP-Complete, but suppose we are interested in determining if G contains a clique of size exactly 4. We can brute force check this by considering every subset of vertices of size 4 in time $\Theta(n^4)$ (e.g., with 4 nested loops), and therefore the problem is no longer NP-Complete if we know we want a clique of size 4. Give an algorithm whose running time is $o(n^4)$ (i.e., strictly better than $\Theta(n^4)$) for this problem.

6 NP-Completeness (18 Points)

Suppose that we are interested in creating an investment portfolio that is well-diversified. That is, suppose there are n assests that we could choose to invest in, but some of the assests may be "too correleated" with each other. In this case we may prefer to invest in at most one of those two investments to ensure that our total selection is diversified. Suppose we have a two-dimensional array A such that $A[i][j] = 1$ if the ith asset is correlated with the jth asset and $A[i][j] = 0$ otherwise. We would like to determine if there exists a set of k assets such that none of them are correlated. Show that this problem is NP-Complete.

7 Max Flow (18 Points)

Consider the following flow network. Let c(i,j) denote the capicy of edge (i,j). Suppose c(s,u) = 9, c(s,v) = 4, c(v,u) = 1, c(u,t) = 8, and c(v,t) = 5. Use Ford-Fulkerson to compute a maximum flow in this flow network. What is a minimum cut of the flow network?



Suppose that someone owns n stores in the San Antonio area and wishes to advertise at m different locations. Each of the stores sells a different type of product, and thus we may want to advertise more than one of the stores at a single location. That being said, we only want to advertise a store at a location if the distance between a store and a location is at most r. Suppose each location j only has capacity for c_j advertisements. Further suppose that each store i has a limit l_i on the number of advertisements can be placed for that store. We are interested in computing the maximum number of advertisements we can place subject to these constraints. Show that this problem can be reduced to computing a maximum flow in an appropriately defined flow network. (You do not need a formal proof as to why it works...describing the construction of the flow network will suffice.)