CS525DA
HW#6

DUE: Tuesday, December 2

1. (6 points) A typical dynamic programming algorithm computes the cost of a solution or establishes the existence of a solution without actually constructing the solution. To see how to construct a solution by using an efficient mechanism which tests for the existence of a solution, solve the following:

You are given a boolean function *BlackBox* of two inputs:

-a list of integers $x_1,\ldots,x_n$,

-an integer q

and you are told that, in time $O(1)$, *BlackBox* will return "true" if and only if there is some subset of $x_1,\ldots,x_n$ whose sum is q. Design an algorithm (a program is not needed) with the same input which will return an actual subset of $x_1,\ldots,x_n$ whose sum is q, if such a subset exists, or else it should return "failure".

For example, *BlackBox*((23,27,41,72,-4,6), 29) would return "true", but your algorithm with the same input would return (27,-4,6). Your algorithm may call *BlackBox* as often as it wishes and it should work in time $O(n)$.

2. (6 points) Let τ_b be the tree produced by a breadth-first search of graph $G=(V,E)$ starting at $\sigma \in V$, and let τ_d be the tree produced by a depth-first search of graph $G=(V,E)$ starting at $\sigma \in V$, where the adjacency lists for each $v \in V$ are the same for each search. Must it be the case that $height(\tau_d) \geq height(\tau_b)$? If you answer yes, then justify your answer. If you answer no, then provide a counterexample. Note that for any tree τ, $height(\tau)$ is the number of edges on a longest path from σ to a leaf of τ.

3. (8 points) Assume that you computed a dfs-tree τ for graph G, and you want to find the shortest cycle in G. Neil makes the following claims (for all graphs G):

- Every cycle of G contains a backedge of τ.
- For any backedge from w to v, the length of the only possible cycle containing edge (v,w) is 1 plus the height of w in τ minus the height of v in τ.
- Thus, to find the length of a shortest cycle of G, you only need to add 1 to the minimum (over all backedges of τ) of the difference between the heights of the vertices of the backedge.

Either prove that Neil is correct or give a counterexample to refute him.

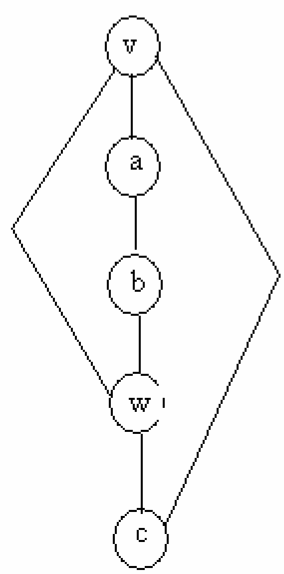
CS525DA
HW#6 SOLUTIONS

1. $S \leftarrow x_1, \ldots, x_n$
   if not *BlackBox*(S,q) **then return** ("failure") $O(1)$
   for $i \leftarrow 1$ **to** n **do** n times
   $S \leftarrow S - x_i$
   if not *BlackBox*(S,q) **then** ▸ we really need x_i ; put it back
   $S \leftarrow S \cup x_i$
   return S $O(1)$

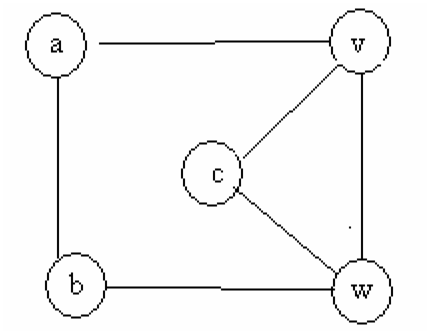
Since the body of the loop is executed in $O(1)$ time, the time to execute the loop (and the program) is $O(n)$.

2. It is always the case that $height(\tau_d) \geq height(\tau_b)$. For any $v \in V$, the height of v in τ_b is the distance σ from to v, and the corresponding path is a shortest path. Since in any tree the path from σ to v is a path in the underlying graph, the path from σ to v in τ_d can not be shorter than a shortest path from σ to v in τ_b. Thus, the height of any vertex in τ_b is less than or equal to its height in τ_d.

3. As usual, Neil is wrong. Consider the dfs tree



for graph



Neil would have us believe that the shortest cycle in G has length 4, whereas it contains a triangle.