GREEDY ALGORITHMS

MATH CIRCLE (HS1) 11/17/2013

Introduction

1) Normal coins in the U.S. come in denominations/values of $1, 5, 10, 25$ cents. What happens if we change these denominations?

Group	Denominations
A	$1, 5, 10, 25$
B	$4, 10, 25$
C	$2, 16, 20$
D	$3, 5, 11, 40$

a) For each group of denominations decide if it is possible to make change, and if so, the fewest number of coins required, for the following amounts of money:

i) 14 ii) 50 iii) 41 iv) 76

b) For each group of denominations, determine all the amounts you can make change for.

c) Try to come up with a method/algorithm for making change with group A. Bonus: Try to do so "locally", i.e. without any planning ahead.

2) Recall that a graph $G = (V, E)$ is a set of vertices (denoted by V) as well as a set of edges (denoted E) between the vertices.

A graph is *connected* if it is possible to travel (along edges) between any two vertices.

A *subgraph* of G is another graph $H = (V', E')$ with $V' = V$ and $E' \subseteq E$; that is, H has the same vertices as G, every edge in H is an edge in G, but not every edge from G is in H. Even more simply, we get subgraphs of G by removing edges from G.

A *minimal spanning tree* (MST) of a connected graph G is a connected subgraph H of G with the smallest possible number of edges.

a) Find a MST for each of the graphs on the next page.

b) Try to find a method/algorithm for coming up with MST for arbitrary connected graphs. Hint: Start at an arbitrary vertex, and build up a minimal spanning tree one edge at a time.

Some problems adapted from previous Math Circle handouts made by Clint Givens.

Graphs:

Definition: A greedy algorithm is an algorithm that makes "locally" optimal choices at each stage, in the hope of ending up with a globally optimal solution.

Greedy Algorithm For Making Change

Making Change Greedy Algorithm: Continually pick the largest denomination possible until you either succeed in making change or you cannot pick any more coins.

3) For each group from Problem 1, decide which statement below is correct (and give a reason for your answer):

i) If it is possible to make change, the algorithm always gives a way to make change optimally (with the fewest coins).

ii) If it is possible to make change, the algorithm always gives a way to make change, but not always optimally.

iii) If it is possible to make change, the algorithm can fail to make change.

4) Suppose we want to add a fifth coin (worth more than 25 cents) to group A. Find a value for this coin so that statement i) is still true. Bonus: Find the smallest value possible.

5) Challenge: a) Suppose that $\{a_1, \ldots, a_n\}$ is a set of denominations so that statement i) is true. Show that statement i) is still true for $\{a_1, \ldots, a_n, 2a_n\}$.

b) Use a) to build a set of denominations so that you can give change for any value less than a dollar using ≤ 7 coins.

Greedy Algorithm For Finding Minimal Weight Spanning Trees

A *weighted* graph is a graph such that each edge has a weight (i.e. positive number) associated with it. The weight of a (weighted) graph is the sum of all its edge's weights.

A *minimal weight spanning tree* (MWST) of a connected graph G is a connected subgraph H of G with the smallest possible weight.

Example: If every edge has weight 1, then MWST are identical to MST.

6) For the graphs on the next page, find a MWST.

7) a) Modify the algorithm from Problem 2b) to get a greedy algorithm for trying to generate MWST. This is called Prim's algorithm.

b) Use your algorithm on the graphs on the next page. Do you think that the algorithm always gives us MWST?

8) Come up with another greedy algorithm for generating MWST. Rough idea: At each stage, add an edge (from anywhere in the graph) with minimal weight. This is called Kruskal's algorithm. Test your algorithm using the graphs on the next page.

9) Use both Prim's algorithm and Kruskal's algorithm on Primland and Kruskalia. Show that they may generate the same MWST, but don't always need to. If they don't generate the same MWST, do you notice any similarities between the MWST?

10) Challenge: Try to prove Kruskal's algorithm always generates MWST. Hint: Prove by induction that at every stage, the edges chosen are part of *some* MWST.