The Greedy Algorithm

TJHSST Senior Computer Team

October 31, 2003

1 Introduction

"Your father wanted you to have this when you were old enough." -Ben Kenobi

You are about embark on a mission, a quest if you will - for an algorithm so powerful, so efficient, so easy and quick to code that it should be a part of any programmer's arsenal. The fundamental method of the greedy algorithm is greed(!) - building a solution by keeping the best result for a smaller problem and disgarding all possibilities. Greedy solutions are fast and easy to code, and usually are O(n) or O(n^2).

2 Building a Solution

"Try not. Code or code not. There is no try." -Yoda

How can we be greedy? The key is building an algorithm correctly and efficiently. Let's start by looking at a fairly straightforward example:

Example 1: Fractional Knapsack [traditional]
A robber enters a 7-11 with a knapsack that can hold up to n pounds of merchandise. There are k boxes of doughnuts in the store, and for each box k he can take all the doughnuts, or he can take doughnuts individually without taking the entire box. Each box has a different weight, given by w_k and different value, given by v_k. What is the greatest value of merchandise he can steal and and hold in his knapsack? Note: Since he is a robber and is not confined by the bounds of human decency, it IS possible for him to split up a doughnut, that monster.

Let's say you are given the following list of items and assume that you can hold up to 10 lbs.:

k	1	2	3	4	5
v_k	14	10	9	6	4
w_k	7	6	5	5	4

We don't have the restriction that we must take an entire box, therefore, we want to get the most out of each individual pound, in other words, we want to use the highest value-to-pound ratios we can for each pound. Sorting the list by the ratio of value to weight, we have

k	1	3	2	4	5
v_k	14	9	10	6	4
w_k	7	5	6	5	4
v/w	2.0	1.8	1.667	1.2	1.0

For our greedy solution, we go straight through the list, taking as many doughnuts as we can starting from those with the highest value-to-weight ratio. In any sort of fractional knapsack problem such as this, we sort by ratio of value to weight, and take from the best choice until we've filled the knapsack up. We know this will work because we are guaranteeing we get the most value for the pound.

3 Trouble Ahead!

> "Sir, the possibility of sucessfully solving all the problems on this contest using Greedy is approximately 3,720 to 1." -C3PO

Often solutions may look like they are Greedy-able but aren't. Let's look at the following example:

Example 2: Friendly Coins [Some European Contest]
Given the denominations of coins for a newly founded country, the Dairy Republic, and some monetary amount, find the smallest set of coins that sums to that amount. The Dairy Republic is guaranteed to have a 1 cent coin.

Greedy Solution: Take the largest coin value that isn't more than the goal and iterate on the total minus this value.

> "The Force is strong with this one." -Vader

This algorithm won't always work as we can see in the following counterexample:
Let the set of demoninations be 1, 5, 8, 10 and a goal of 13. This greedy algorithm would take one 10, and then three 1's, for a total of four coins, when the two coin solution 5, 8 also exists. All hope is not lost, however: this algorithm usually works, in fact it works for the US coin system 1, 5, 10, 25.

4 Practical Points

> "You must unlearn what you have learned." -Yoda

Greedy is not always an algorithm that will get you full credit, but often it will give you most of the test cases on a problem that might be very difficult to solve otherwise, allowing you to move on in a contest. After doing a few practice contests you will hopefully get the feel for when it is time to buckle down for an hour and solve a problem or whether it's time to get Greedy and move on.

> "I would rather get most of the test cases on all of the problems than all of the test cases on some of the problems." -Larry

5 Problems for Practice

Take a look at the following problems. Try to find a greedy algorithm that works, and explain why it will be sucessful all of the time or most of the time.

Problem 1: Mixing Milk [USACO Training Pages]
Since milk packaging is such a low margin business, it is important to keep the price of the raw product (milk) as low as possible. Help Merry Milk Makers get the milk they need in the cheapest possible manner.
The Merry Milk Makers company has several farmers from which they may buy milk, and each one has a (potentially) different price at which they sell to the milk packing plant. Moreover, as a cow can only produce so much milk a day, the farmers only have so much milk to sell per day. Each day, Merry Milk Makers can purchase an integral amount of milk from each farmer, less than or equal to the farmer's limit.
Given the Merry Milk Makers' daily requirement of milk, along with the cost per gallon and amount of available milk for each farmer, calculate the minimum amount of money that it takes to fulfill the Merry Milk Makers' requirements.
Note: The total milk produced per day by the farmers will be sufficient to meet the demands of the Merry Milk Makers.

Problem 2: Barn Repair [1999 USACO Spring Open]
There is a long list of stalls, some of which need to be covered with boards. You can use up to N (1 ¡= N ¡= 50) boards, each of which may cover any number of consecutive stalls. Cover all the necessary stalls, while covering as few total stalls as possible.