Greedy Algorithms

Mashiwat Tabassum Waishy

Lecturer

Department of CSE

STAMFORD UNIVERSITY BANGLADESH

Greedy Introduction

- ❑ Greedy algorithms are simple and straightforward.
- ❑ They are shortsighted in their approach
- ❑ A greedy algorithm is similar to a dynamic programming algorithm, but the difference is that solutions to the sub problems do not have to be known at each stage
- ❑ It is used to solve the optimization problems.

Optimization Problems

- ❑ An optimization problem is one in which you want to find, not just a solution, but the best solution
- ❑ A “greedy algorithm” sometimes works well for optimization problems
- ❑ A greedy algorithm works in phases. At each phase:
  - ▪ You take the best you can get right now, without regard for future consequences
  - ▪ You hope that by choosing a local optimum at each step, you will end up at a global optimum.

Optimization Problems

- A problem that may have many feasible solutions.
- Each solution has a value.
- In maximization problem, we wish to find a solution to maximize the value
- In the minimization problem, we wish to find a solution to minimize the value.

Technique to Solve a Problem

- Greedy Method
- Dynamic Programming
- Branch and Bound

Greedy Algorithms

- Many optimization problems can be solved using a greedy approach
  - The basic principle is that local optimal decisions may be used to build an optimal solution
  - But the greedy approach may not always lead to an optimal solution overall for all problems
  - The key is knowing which problems will work with this approach and which will not.

Greedy algorithms

- A ***greedy algorithm*** always makes the choice that looks best at the moment.
  - My everyday examples:
    - Driving in Los Angeles, NY, or Boston for that matter
    - Playing cards
    - Invest on stocks
    - Choose a university
  - The hope: a locally optimal choice will lead to a globally optimal solution.
  - For some problems, it works.
- Greedy algorithms tend to be easier to code.

Greedy Technique

- Greedy algorithms are simple and straightforward.
- They are shortsighted in their approach in the sense that they take decisions on the basis of information at hand without worrying about the effect these decisions may have in the future.
- They are easy to invent, easy to implement and most of the time quite efficient.
- Many problems cannot be solved correctly by greedy approach. Greedy algorithms are used to solve optimization problems

Greedy Approach

- Greedy Algorithm works by making the decision that seems most promising at any moment; it never reconsiders this decision, whatever situation may arise later.

Algorithm

```
Algorithm Greedy(a, n){
       for i=1 to n do
       X = Select(a);
         If feasible(x) then
           Solution = Solution + x;
       }
```

A simple example

- Problem: Pick k numbers out of n numbers such that the sum of these k numbers is the largest.
- Algorithm:

```
FOR i = 1 to k
    pick out the largest number and
    delete this number from the
    input.
END FOR
```

The greedy method

- Suppose that a problem can be solved by a sequence of decisions. The greedy method has that each decision is locally optimal. These locally optimal solutions will finally add up to a globally optimal solution.

- Only a few optimization problems can be solved by the greedy method.

Another Example

- Greedy Algorithm works by making the decision that seems most promising at any moment; it never reconsiders this decision, whatever situation may arise later.
- As an example consider the problem of "Making Change".
- Coins available are:
  - dollars (100 cents)
  - quarters (25 cents)
  - dimes (10 cents)
  - nickels (5 cents)
  - pennies (1 cent)

Making Change Problem

- **Problem** Make a change of a given amount using the smallest possible number of coins.
- **Informal Algorithm**
  - Start with nothing.
  - at every stage without passing the given amount.
    - add the largest to the coins already chosen.

Coin Changing Problem

Algorithm:

Make change for n units using the least possible number of coins.

```
MAKE-CHANGE (n)
    $C \leftarrow \{100, 25, 10, 15, 1\}$          // constant.
    $S \leftarrow \{\}$;                    // set that will hold the solution set.
    $Sum \leftarrow 0$ sum of item in solution set
    WHILE sum not = n
        x = largest item in set C such that $sum + x \le n$
        IF no such item THEN
            RETURN    "No Solution"
        $S \leftarrow S$ {value of x} & Remove x from C.
        $sum \leftarrow sum + x$
    RETURN S
```

Features of Problems solved by Greedy Algorithms

- To construct the solution in an optimal way. Algorithm maintains two sets. One contains chosen items and the other contains rejected items.
- The greedy algorithm consists of four (**4**) function.
  - **A Candidate Set:-** Solution is created from this set.
  - **A Selection Set:-** A Function used to chose the best candidate to be added to the solution.
  - **A Feasibility Set:-** A function that checks the **feasibility of a set**.
  - **A Objective Function:-** A Function which is used to assign value to a solution or partial solution.
  - A Solution Function:- A Function which is used to indicate whether a complete solution has been reached.

Structure of Greedy Algorithm

- Initially the set of chosen items is empty i.e., solution set.
- At each step
    - item will be added in a solution set by using selection function.
    - IF the set would no longer be feasible
        - reject items under consideration (and is never consider again).
    - ELSE IF set is still feasible THEN
        - add the current item.

Definition of Feasibility

- A feasible set (of candidates) is promising if it can be extended to produce not merely a solution, but an optimal solution to the problem. In particular, the empty set is always promising why? (because an optimal solution always exists)
- Unlike Dynamic Programming, which solves the subproblems bottom-up, a greedy strategy usually progresses in a top-down fashion, making one greedy choice after another, reducing each problem to a smaller one.

- **Greedy-Choice Property**
  - The "greedy-choice property" and "optimal substructure" are two ingredients in the
    problem that lend to a greedy strategy.
  - It says that a globally optimal solution can be arrived at by making a locally optimal choice.

Greedy Property

It consists of two property.

1. Greedy-Choice Property:

It says that a globally optimal solution can be arrived at by making a locally optimal choice.

2. Optimal Substructure:

An optimal global solution contains the optimal solutions of all its sub problems.

Advantages And Disadvantages

Advantages:

❑ They are easier to implement.

❑ They require much less computing resources.

❑ They are much faster to execute.

❑ Greedy algorithms are used to solve optimization problems.

Disadvantages:

❑ Their only disadvantage being that they not always reach the global optimum solution.

❑ On the other hand, even when the global optimum solution is not reached, most of the times the reached sub-optimum solution is a very good solution.

Thank you