13-Greedy Algorithms

Algorithms that make the choice that looks best at the moment, regardless of future choices.

Examples:
- Making Change
- Activity Selection
- Huffman Codes
- Fractional Knapsack
- Minimum Spanning Tree
Activity Selection

Our goal is to find the largest set of non-conflicting activities.
1) Sort by end-time

2) Take earliest-ending activity that doesn't conflict with your set

3) Repeat until no activities left
Greedy Strategy

1) Cast the problem as one in which we make a choice and have one subproblem to solve.

2) Prove that the greedy choice is always safe, and that it can be combined with the solution to the subproblem for an optimal solution.

When does greedy work?

1) Optimal Substructure

2) Greedy-choice property
Optimal Prefix Codes

i am mason \[ \frac{8 \text{ bits}}{\text{char}} \]
10 char = 80 bits
Huffman Codes

a  b  c  d  e  f  g  h  i
5  1  3  2  1  5  9  3  6  8