Properties of Greedy Algorithm

Problems that can be solved by greedy algorithms have two main properties:

Optimal Substructure The optimal solution to the problem contains within it optimal solutions to subproblems.

Greedy Choice Property A global optimal solution can be arrived at by selecting a local optimal solution.

Difference from DP: No overalapping subproblems.

Activity Selection Problem

Input Set S of n activities $a_1, a_2, \ldots, a_n$.

- s_i = start time of activity i
- f_i = finish time of activity i

Output Subset A of maximum number of compatible activities.

Two activities are compatible if they do not overlap.

i	1	2	3	4	5	6	7	8	9
s_i	1	2	4	1	5	8	9	11	13
f_i	3	5	7	8	9	10	11	14	16

Assume activites are sorted by finishing times.

$$f_1 \leq f_2 \leq \ldots \leq f_n$$

Suppose an optimal solution includes activity a_k.

- This generates two subproblems
  - Selecting from $a_1, \ldots, a_{k-1}$ activites compatible with one another, and that finish before a_k starts (compatible with a_k)
  - Selecting from $a_{k+1}, \ldots, a_n$ activites compatible with one another, and that start after a_k finishes.
- The solutions to the two subproblems must be optimal.
  - Prove using the cut-and-paste approach.

Optimal Substructure

$S_{ij} =$ subset of activites in S that start after a_i finishes and finish before a_j starts

$$\begin{aligned} \text{Let } A_{ik} = A_{ij} \cap S_{ik} \text{ and } A_{kj} &= A_{ij} \cap S_{kj} \\ A_{ij} &= A_{ik} \cup \{a_k\} \cup A_{kj} \\ |A_{ij}| &= |A_{ik}| + |A_{kj}| + 1 \end{aligned}$$

$c[i,j] =$ size of max-size subset of mutually compatible activites in S_{ij}

$$\text{Recursive Solution} \quad c[i,j] = \begin{cases} 0 & \text{if } S_{ij} = \emptyset \\ \max\limits_{a_k \in S_{ij}} \{c[i,k] + c[k,j] + 1\} & \text{if } S_{ij} \neq \emptyset \end{cases}$$

After you choose a_k, the subproblem focuses on the set of

$$S_k = \{a_i \in S : s_i \geq f_k\}$$

Theorem If S_k is nonempty and a_m has the earliest finish time in S_k, then a_m is included in some optimal solution.

Proof Let A_k be an optimal solution to S_k, and let a_j have the earliest finish time of any activity in A_k. If $a_j = a_m$, done. Otherwise, let $A'_k = A_k - \{a_j\} \cup \{a_m\}$ be A_k but with a_m substituted for a_j.

```
Recursive-Activity-Selector(s, f, k, n)
    m = k + 1
    while m <= n and s[m] < f[k]
        m = m + 1
    if m <= n
        return {m} U Recursive-Activity-Selector(s, f, m, n)
    else
        return {}
```

Typical Steps of Greedy Algorithm

- Cast the optimization problem as one in which we make a (greedy) choice and are left with one subproblem to solve.
- Prove that there's always an optimal solution to the original problem that makes the greedy choice, so that the greedy choice is always safe.
- Show that greedy choice and optimal solution to the subproblem $\Rightarrow$ optimal solution to the problem.
- Make the greedy choice and solve *top-down*.