Greedy Algorithms
Greedy

Idea

- Build solution by making a sequence of choices.
  - In each step take the locally best choice.
  - Solve remaining subproblem.

Be careful!

- This procedure does not always yield the globally optimal solution.
- One has to prove it for each specific problem.
Examples of Greedy algorithms

- Kruskal
- Huffman-Codes
- Problems with a matroid structure (see Cormen for details)
Greedy vs. Dynamic Programming (DP)

**Optimal Substructure**
- Greedy and DP algorithms make use of the **optimal substructure** of the problem.
- Optimal substructure: Optimal solution to subproblems & locally optimal choice yield a globally optimal solution of the problem.

**DP**
Make optimal choice **depending** on the solutions of the subproblems.

**Greedy**
Make optimal choice **independent** of the solutions of the subproblems.
Greedy-Algorithm Design

Basic Steps (Cormen)

1. Determine the optimal substructure of the problem.
2. Develop a recursive algorithm.
3. Prove that at any stage the greedy-choice is locally optimal and leads to a globally optimal solution.
4. Convert the recursive algorithm to an iterative algorithm.
Recursive Kruskal

Input: \( G = (V,E,l) \), \( F \) (initially empty)
Output: \( T = (V,F) \)

If \( E = \emptyset \) return \( (V,F) \)

Find minimal edge \( e \) such that \( F \cup e \) contains no cycle.

\( F := F \cup e \)
\( G := (V,E\setminus e,l) \)

return Kruskal(G,F)