2017-10-12

- **Principle of optimality**: any subpolicy of an optimum policy must itself be an optimum policy with regard to the initial and terminal states of the subpolicy.
- Two key properties of DP for optimization: (1) overlapping subproblems, (2) optimal substructure.
- Two approaches of dynamic programming in terms of subproblem graph:
  - **Top-down with memoization**: depth-first search.
  - **Bottom-up with tabulation**: reverse topological search.
- Rod cutting problem:
  - Input: a rod of length n and a table of prices p_i for $i = 1, 2, \ldots, n$.
  - Output: the maximum revenue r_n obtainable by cutting up the rod and selling the pieces.
  - Recursion: $r_i = \begin{cases} 0 \text{ if } i = 0 \\ \max_{1 \le j \le i} p_j + r_{i-j} \text{ if } i \ge 1 \end{cases}$
- Stamp problem:
  - Input: the postage n and the stamps with values v_i for $i = 1, 2, \ldots, k$.
  - Output: the minimum number of stamps S_n to cover the postage.
  - Recursion: $S_i = \begin{cases} 0 \text{ if } i = 0 \\ \max_{1 \le j \le k} 1 + S_{i-v_j} \text{ if } i \ge 1 \end{cases}$
- Matrix-chain multiplication:
  - Input: a sequence of n matrices $A_1, A_2, \ldots, A_n$ and the corresponding sequence $l_0, l_1, \ldots, l_n$ indicating the dimensionality of As.
  - Output: a order of matrix multiplications with the minimum number of operations $M_{1,n}$ to obtain the product of $A_1 A_2 \ldots A_n$.
  - Recursion: $M_{i,j} = \begin{cases} 0 \text{ if } i \ge j \\ \min_{i \le k < j} M_{i,k} + M_{k_1,j} + l_{i-1} l_k l_j \text{ if } i < j \end{cases}$
- Sequence alignment problem:
  - Input: two sequences $X = x_1 x_2 \ldots x_m$ and $Y = y_1 y_2 \ldots y_n$, and the cost of insertion C_i, deletion C_d and substitutions C_s.
  - Output: the minimal cost M for aligning two sequences.
  - Recursion: $M_{i,j} = \begin{cases} jC_i \text{ if } i = 0 \\ iC_d \text{ if } j = 0 \\ \min\{M_{i-1,j-1} + C_s, M_{i-1,j} + C_d, M_{i,j-1} + C_i\} \text{ otherwise} \end{cases}$