2017-10-19

- Sequence alignment problem:
    - Input: two sequences $X = x1x2...xm$ and $Y = y1y2...yn$, and the cost of insertion C_i, deletion C_d and substitutions C_s.
    - Output: the minimal cost M for aligning two sequences.
    - Recursion: $M_{i,j} = \begin{cases} jC_i \text{ if } i = 0 \\ iC_d \text{ if } j = 0 \\ \min\{M_{i-1,j-1} + C_s, M_{i-1,j} + C_d, M_{i,j-1} + C_i\} \text{ otherwise} \end{cases}$
    - Time complexity: $T(m, n) = \Theta(mn)$.
    - Space complexity: $S(m, n) = \Theta(mn)$ or $S(m, n) = \Theta(n)$ but the solution cannot be reconstructed.
- Space-efficient algorithm for sequence alignment problem:
    - Divide and conquer + Dynamic programming.
    - Find the min-cost alignment → Find the shortest path.
    - Solution: $F_{m,n} = \min_{0 \le u \le m} F_{u,v} + B_{u,v} \;\; \forall v$.
        - $F_{i,j}$: length of the shortest path from $(0, 0)$ to (i, j).
        - $B_{i,j}$: length of the shortest path from (i, j) to (m, n).
    - Optimal solution:
        - $v = n/2$
        - $u^* = \arg\min_{0 \le u \le m} F_{u,v} + B_{u,v}$
    - Time complexity: $T(m, n) = \begin{cases} O(m) \text{ if } n = 1 \\ T(u^*, n/2) + T(m - u^*, n/2) + O(mn) \text{ if } n > 1 \end{cases} \implies T(m, n) = O(mn)$.
- Weighted interval scheduling problem:
    - Input: n job requests with start times s, finish times f.
    - Output: the maximum number of compatible jobs.
    - $p(j)$ = largest index $i < j$ s.t. jobs i and j are compatible.
    - Sort the requests in non-decreasing order.
    - Recursion: $M_i = \begin{cases} 0 \text{ if } i = 0 \\ \max\{v_i + M_{p(i)}, M_{i-1}\} \text{ otherwise} \end{cases}$
    - Time complexity: $T(n) = \Theta(n)$.
- Knapsack problem:
    - Input: n items where i-th item has value v_i and weighs w_i (v_i and w_i are positive integers).
    - Output: the maximum value for the knapsack with capacity of W.
    - 0-1: $M_{i,w} = \begin{cases} 0 \text{ if } i = 0 \\ M_{i-1,w} \text{ if } w_i > w \\ \max\{v_i + M_{i-1,w-w_i}, M_{i-1,w}\} \text{ otherwise} \end{cases}$
    - Unbounded: $M_w = \begin{cases} 0 \text{ if } w = 0 \text{ or } w_i > w \\ \max_{1 \le i \le n, w_i \le w}(v_i + M_{w-w_i}) \text{ otherwise} \end{cases}$

- Multidimensional: $M_{i,w,d} = \begin{cases} 0 \text{ if } i = 0 \\ M_{i-1,w,d} \text{ if } w_i > w \\ \max\{v_i + M_{i-1,w-w_i,d-d_i}, M_{i-1,w,d}\} \text{ otherwise} \end{cases}$
- Multiple-choice: $M_{i,w} = \begin{cases} 0 \text{ if } i = 0 \\ M_{i-1,w} \text{ if } w_{i,j} > w \ \forall j \\ \max_{1 \le j \le n_i}\{v_{i,j} + M_{i-1,w-w_{i,j}}, M_{i-1,w}\} \text{ otherwise} \end{cases}$
- Fractional: by greedy algorithm.