Dynamic Programming

Part II

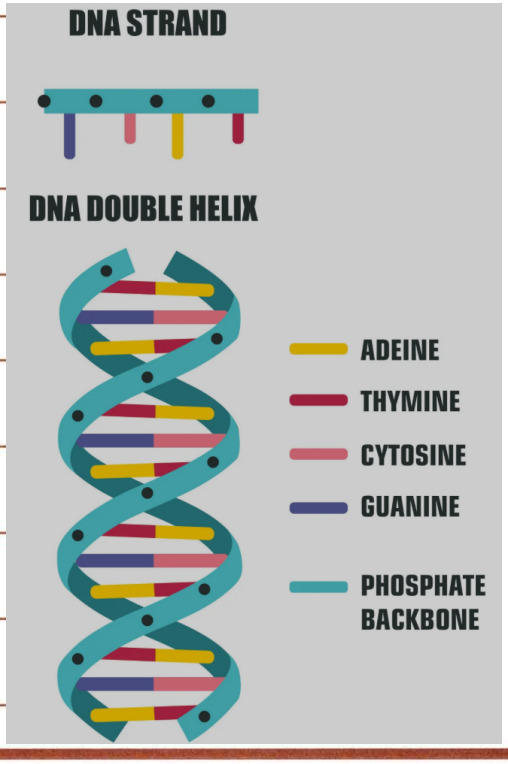
Reminder:

General Approach to Solving Optimization Problems Using Dynamic Programming

1. Characterize the structure of an opt. solution
2. Recursively define the value of an opt. solution
3. Compute the value of an opt. solution in a bottom up fashion
4. Construct an opt. solution from computed information

Sequence Alignment Problem

Background:
- A DNA strand consists of molecules called bases.



Ex.

S_1 = ACCGGTCG

S_2 = CCAGGTGGC

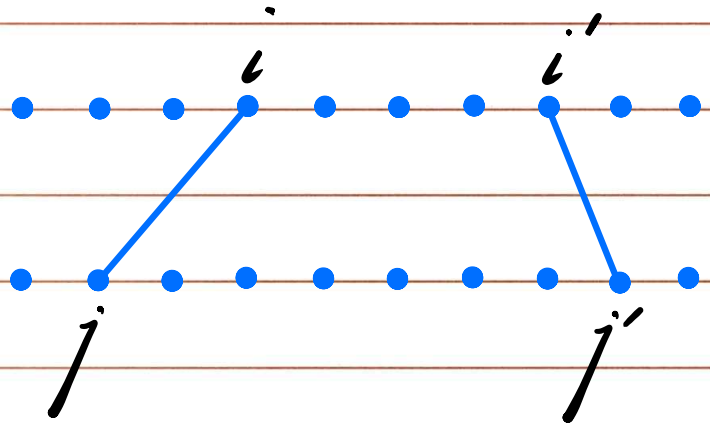
We need to come up with a concise definition of similarity between two strings.

Def. Given 2 strings X and Y,

$$X = \{x_1, x_2 \cdots x_m\}$$
$$Y = \{y_1, y_2 \cdots y_n\},$$

a matching is a set of ordered pairs with property that each item occurs at most once.

Def. A matching is an alignment if there are no crossing pairs.



In other words, in an alignment M, if $(i,j), (i',j') \in M$, and $i < i'$, then $j < j'$.

Cost of an alignment

For an alignment M between X and Y,

1- We incur a gap penalty of δ for each gap.

2- For each mismatch (of letters p and q) we incur a mismatch cost of α_{pq}.

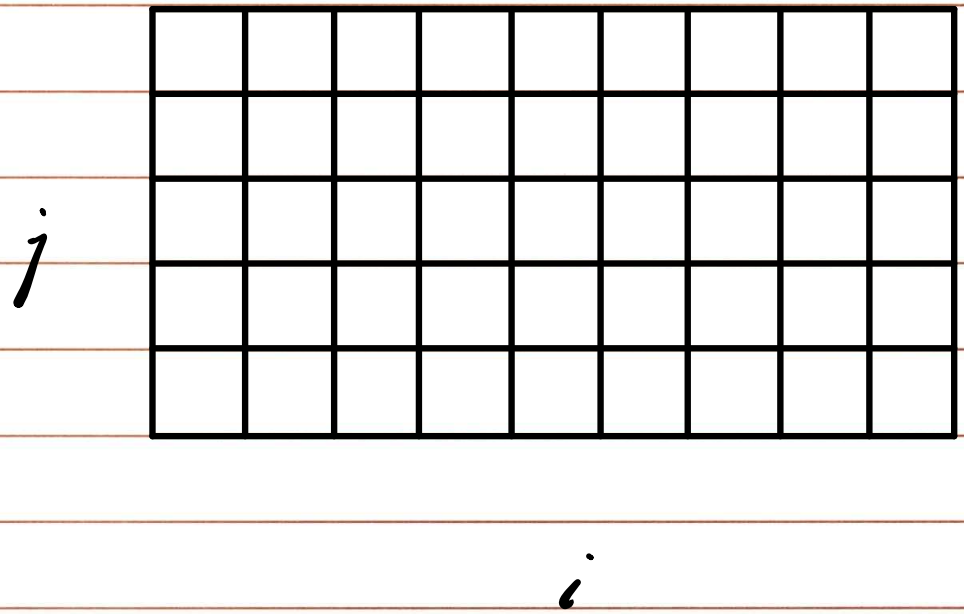
Def. Similarity between strings X and Y, is the minimum cost of an alignment between X and Y.

$$X = \{x_1, x_2 \ \dots \ x_m\}$$

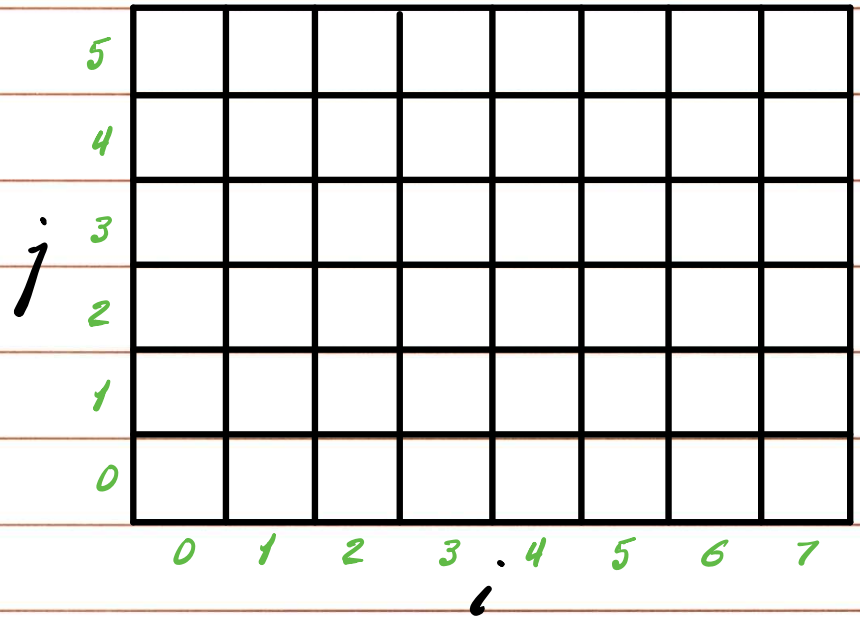
$$Y = \{y_1, y_2 \ \dots \ y_n\}$$

Say M is an opt. solution. Then either $(x_m, y_n) \in M$ or $(x_m, y_n) \notin M$

Bottom up pass:

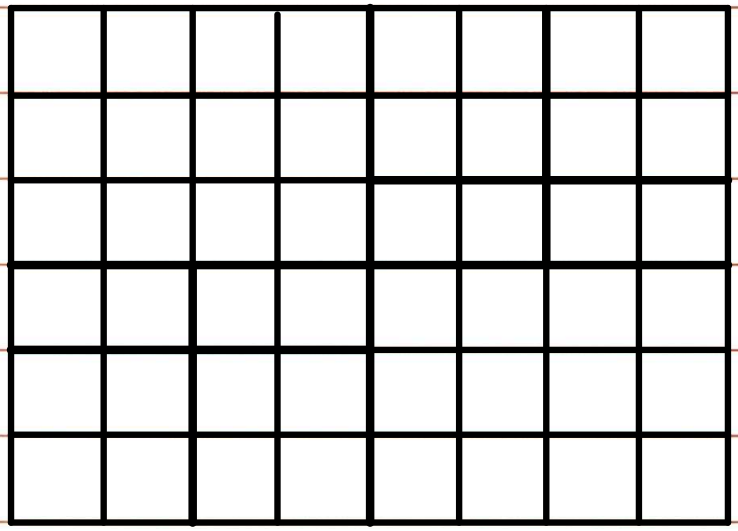


Top_down pass:



Memory Efficient Solution

Complexity Analysis



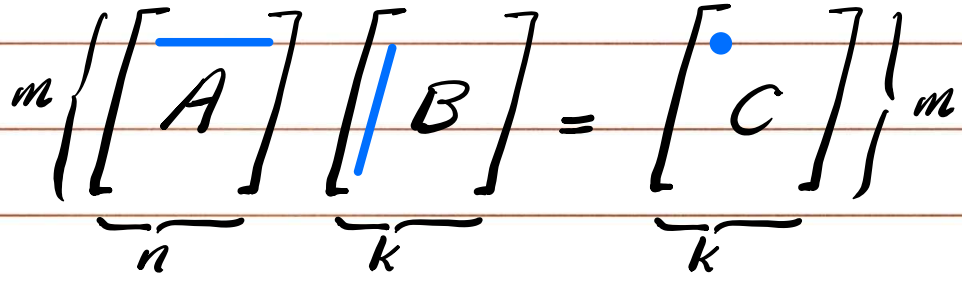
Matrix Chain Multiplication

Problem Statement

Given a sequence of n dense matrices along with their dimensions, find the order of matrix multiplications that minimizes the total number of element multiplications.

$$A = A_1 \cdot A_2 \cdot A_3 \ \ldots \ A_n$$

Recall the no. of element multiplications for two rectangular dense matrices



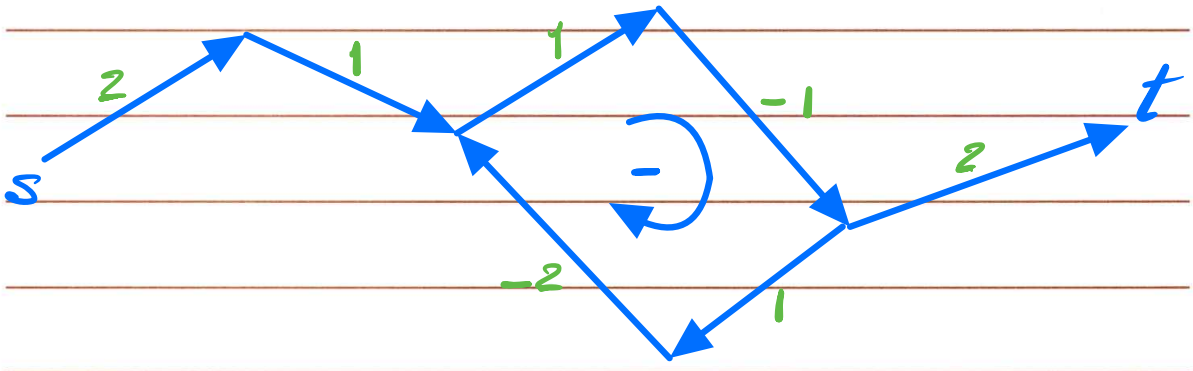
Requires mnk element multiplications.
(for the basic method – not using Strassen's alg.)

Ex.:

B.C.D

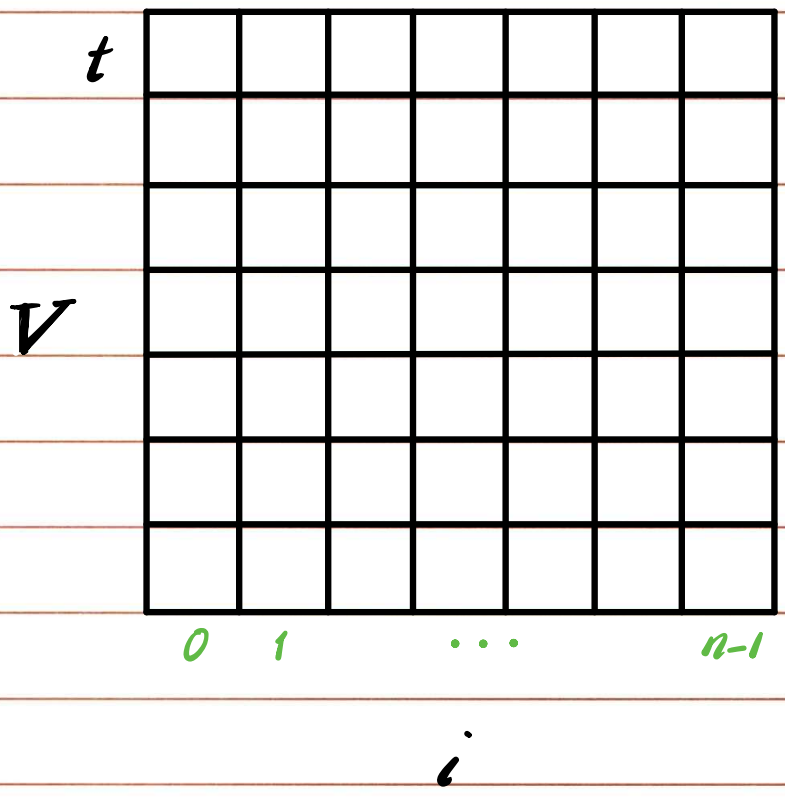
- B is 2x10
- C is 10x50
- D is 50x20

Shortest Path Problem
(Dynamic Programming)



What is the shortest distance from S to T?

If G has no negative cycles, then there is a shortest path from s to t that is simple and hence has at most $n-1$ edges.


t
V
0
1
...
n-1
i

Bellman-Ford Alg.

Shortest Path (G, s, t)

n = no. of nodes in G

Define $OPT(0,t) = 0$, $OPT(0,v) = \infty$

for $i = 1$ to $n-1$

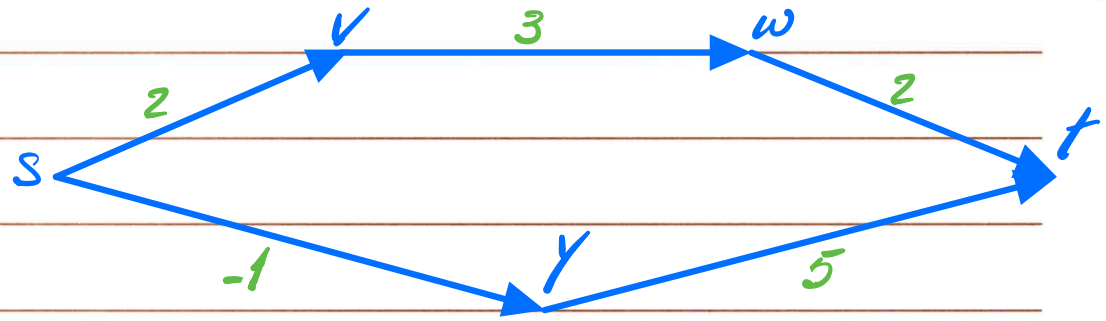
for $v \in V$ in any order

Use recurrence formula 3

endfor

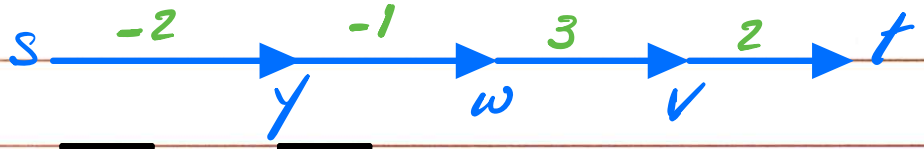
endfor

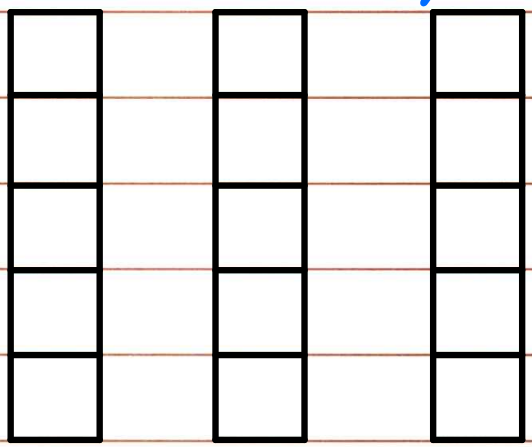
Ex.

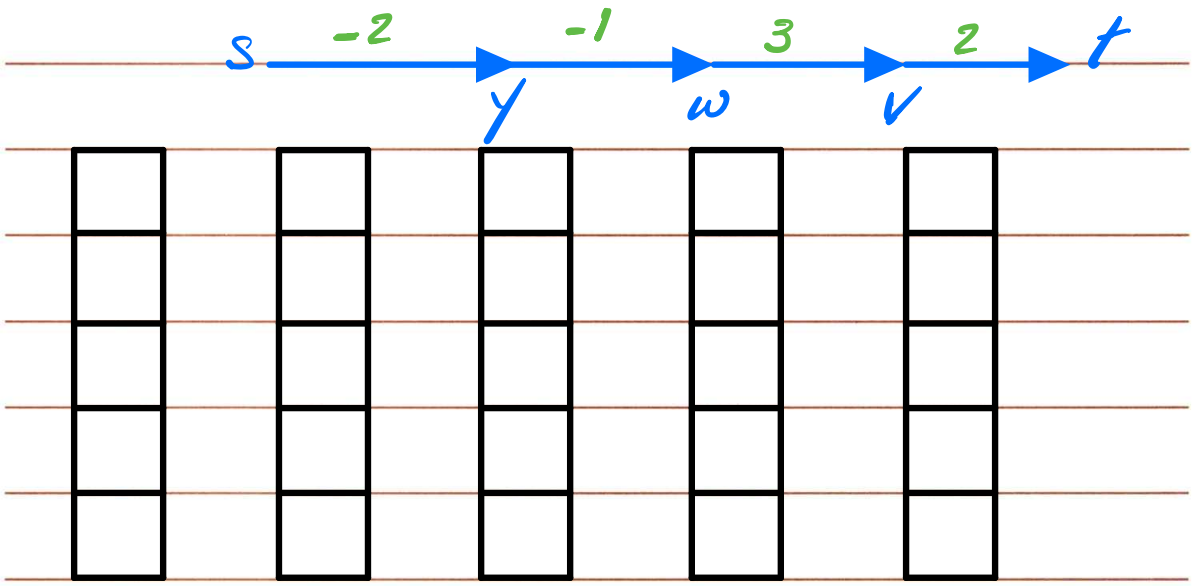


t					
y					
v					
w					
s					
	0	1	2	3	4

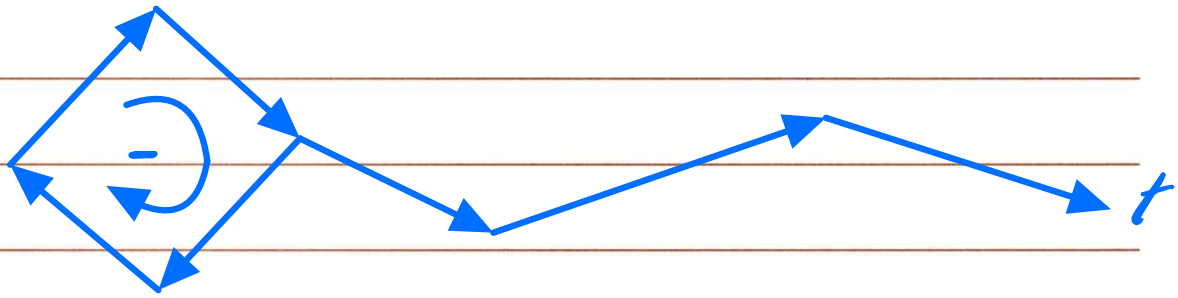
what happens if we only use one column?






s
-2
y
-1
w
3
v
2
t

How can we detect if a directed graph has a negative cycle?



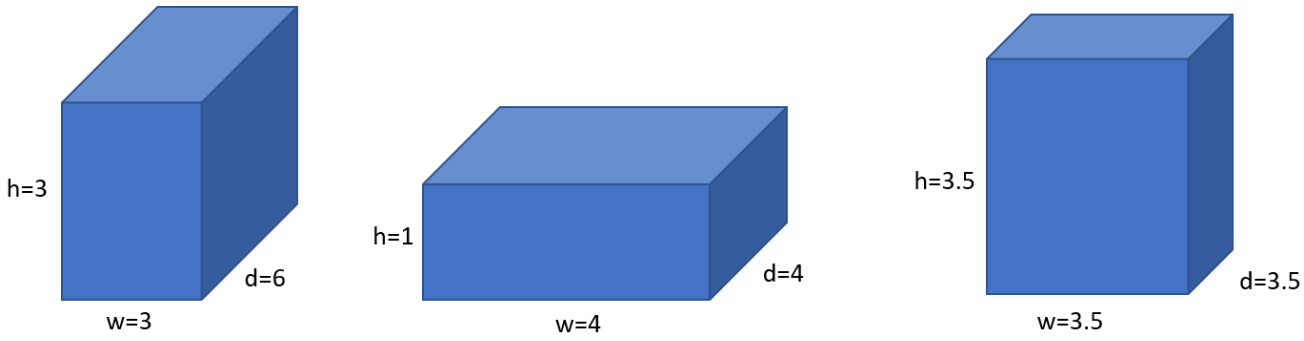
Bellman-Ford

Dijkstra

1. When their respective sport is not in season, USC's student-athletes are very involved in their community, helping people and spreading goodwill for the school. Unfortunately, NCAA regulations limit each student-athlete to at most one community service project per semester, so the athletic department is not always able to help every deserving charity. For the upcoming semester, we have S student-athletes who want to volunteer their time, and B buses to help get them between campus and the location of their volunteering. There are F projects under consideration; project i requires s_i student-athletes and b_i buses to accomplish, and will generate $g_i > 0$ units of goodwill for the university. Our goal is to maximize the goodwill generated for the university subject to these constraints. Note that each project must be undertaken entirely or not done at all -- we cannot choose, for example, to do half of project i to get half of g_i goodwill.

2. Suppose you are organizing a company party. The corporation has a hierarchical ranking structure; that is, the CEO is the root node of the hierarchy tree, and the CEO's immediate subordinates are the children of the root node, and so on in this fashion. To keep the party fun for all involved, you will not invite any employee whose immediate superior is invited. Each employee j has a value v_j (a positive integer), representing how enjoyable their presence would be at the party. Our goal is to determine which employees to invite, subject to these constraints, to maximize the total value of invitees.

3. You are given a set of n types of rectangular 3-D boxes, where the i^{th} box has height h(i), width w(i) and depth d(i) (all real numbers). You want to create a stack of boxes which is as tall as possible, but you can only stack a box on top of another box if the dimensions of the 2-D base of the lower box are each strictly larger than those of the 2-D base of the higher box. Of course, you can rotate a box so that any side functions as its base. It is also allowable to use multiple instances of the same type of box.


h=3
d=6
w=3
h=1
d=4
w=4
h=3.5
d=3.5
w=3.5