

Module 4

Dynamic Programming

Dr. Natarajan Meghanathan
Professor of Computer Science
Jackson State University
Jackson, MS 39217
E-mail: natarajan.meghanathan@jsums.edu

Introduction to Dynamic Programming

- Dynamic Programming is a general algorithm design technique for solving problems defined by recurrences with overlapping sub problems
- “Programming” here means “planning”
- Main idea:
 - set up a recurrence relating a solution to a larger instance to solutions of some smaller instances
 - solve smaller instances once
 - record solutions in a table
 - extract solution to the initial instance from that table
 - Dynamic programming can be interpreted as a special variety of space-and-time tradeoff (store the results of smaller instances and solve a larger instance more quickly rather than repeatedly solving the smaller instances more than once).
- Example: Fibonacci series 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55
- $F(n) = F(n-1) + F(n-2)$, for $n > 1$. $F(0)=0$; $F(1) = 1$
 - $F(6) = F(5) + F(4)$.
 - $F(5) = F(4) + F(3)$. Note that we do not solve $F(4)$ twice. We find $F(4)$ only once and use that to compute $F(5)$ and $F(6)$.

Coin-Collecting Problem

- **Problem Statement**: Several coins are placed in cells of an $n \times m$ board, no more than one coin per cell. A robot, located in the upper left cell of the board, needs to collect as many of the coins as possible and bring them to the bottom right cell. On each step, the robot can move either one cell to the right or one cell down from its current location. When the robot visits a cell with a coin, it always picks up that coin. Design an algorithm to find the maximum number of coins the robot can collect and a path it needs to follow to do this.
- **Solution**: Let $F(i, j)$ be the largest number of coins the robot can collect and bring to the cell (i, j) in the i th row and j th column of the board. It can reach this cell either from the adjacent cell $(i-1, j)$ above it or from the adjacent cell $(i, j-1)$ to the left of it.
- The largest numbers of coins that can be brought to these cells are $F(i-1, j)$ and $F(i, j-1)$ respectively. Of course, there are no adjacent cells to the left of the first column and above the first row. For such cells, we assume there are 0 neighbors.
- Hence, the largest number of coins the robot can bring to cell (i, j) is the maximum of the two numbers $F(i-1, j)$ and $F(i, j-1)$, plus the one possible coin at cell (i, j) itself.

Coin-Collecting Problem

Recurrence

$$F(i, j) = \max\{F(i - 1, j), F(i, j - 1)\} + c_{ij} \quad \text{for } 1 \leq i \leq n, \quad 1 \leq j \leq m$$

$$F(0, j) = 0 \quad \text{for } 1 \leq j \leq m \quad \text{and} \quad F(i, 0) = 0 \quad \text{for } 1 \leq i \leq n,$$

where $c_{ij} = 1$ if there is a coin in cell (i, j) and $c_{ij} = 0$ otherwise.

ALGORITHM *RobotCoinCollection*($C[1..n, 1..m]$)

//Applies dynamic programming to compute the largest number of
//coins a robot can collect on an $n \times m$ board by starting at $(1, 1)$
//and moving right and down from upper left to down right corner

//Input: Matrix $C[1..n, 1..m]$ whose elements are equal to 1 and 0

//for cells with and without a coin, respectively

//Output: Largest number of coins the robot can bring to cell (n, m)

$F[1, 1] \leftarrow C[1, 1]$; for $j \leftarrow 2$ to m do $F[1, j] \leftarrow F[1, j - 1] + C[1, j]$

for $i \leftarrow 2$ to n do

$F[i, 1] \leftarrow F[i - 1, 1] + C[i, 1]$

 for $j \leftarrow 2$ to m do

$F[i, j] \leftarrow \max(F[i - 1, j], F[i, j - 1]) + C[i, j]$

return $F[n, m]$

Time Complexity: $\Theta(nm)$ Space Complexity: $\Theta(nm)$

Coin-Collecting Problem

- Tracing back the optimal path:
- It is possible to trace the computations backwards to get an optimal path.
- If $F(i-1, j) > F(i, j-1)$, an optimal path to cell (i, j) must come down from the adjacent cell above it;
- If $F(i-1, j) < F(i, j-1)$, an optimal path to cell (i, j) must come from the adjacent cell on the left;
- If $F(i-1, j) = F(i, j-1)$, it can reach cell (i, j) from either direction. Ties can be ignored by giving preference to coming from the adjacent cell above.
- If there is only one choice, i.e., either $F(i-1, j)$ or $F(i, j-1)$ are not available, use the other available choice.
- The optimal path can be obtained in $\Theta(n+m)$ time.

Coin-Collecting Problem: Ex-1

	1	2	3	4	5	6
1					5	
2		4		3		
3				2		7
4			8			2
5	9				6	














	1	2	3	4	5	6
1	0	0	0	0	5	5
2	0	4	4	7	7	7
3	0	4	4	9	9	16
4	0	4	12	12	12	18
5	9	9	12	12	18	18

Coin-Collecting Problem: Ex-1 (1)

	1	2	3	4	5	6
1	0	0	0	0	5	5
2	0	4	4	7	7	7
3	0	4	4	9	9	16
4	0	4	12	12	12	18
5	9	9	12	12	18	18

	1	2	3	4	5	6
1					5	
2		4		3		
3				2		7
4			8			2
5	9				6	

Coin-Collecting Problem: Ex-2

	1	2	3	4	5	6
1		7 				 4
2			 5	 3		
3		 8				 2
4	 4		 6		 1	
5	 9			 5		
6		 3			 7	

	1	2	3	4	5	6
1	0	7	7	7	7	11
2	0	7	12	15	15	15
3	0	15	15	15	15	17
4	4	15	21	21	22	22
5	13	15	21	26	26	26
6	13	18	21	26	33	33

Coin-Collecting Problem: Ex-2 (1)

	1	2	3	4	5	6
1	0	7	7	7	7	11
2	0	7	12	15	15	15
3	0	15	15	15	15	17
4	4	15	21	21	22	22
5	13	15	21	26	26	26
6	13	18	21	26	33	33

	1	2	3	4	5	6
1		7				4
2			5	3		
3		8				2
4	4		6		1	
5	9			5		
6		3			7	

Computing a binomial coefficient

Binomial coefficients are coefficients of the binomial formula:

$$(a + b)^n = C(n,0)a^n b^0 + \dots + C(n,k)a^{n-k}b^k + \dots + C(n,n)a^0 b^n$$

Recurrence: $C(n,k) = C(n-1,k) + C(n-1,k-1)$ for $n > k > 0$



$$C(n,0) = 1, \quad C(n,n) = 1 \text{ for } n \geq 0$$

Value of $C(n,k)$ can be computed by filling a table:

	0	1	2	...	$k-1$	k
0	1					
1	1	1				
.						
.						
.						
$n-1$					$C(n-1,k-1)$	$C(n-1,k)$
n						$C(n,k)$

$${}^n C_k = \frac{n!}{k! * (n-k)!}$$

Computing $C(12,5)$

		k 					
		0	1	2	3	4	5
n 	0	1					
	1	1	1				
	2	1	2	1			
	3	1	3	3	1		
	4	1	4	6	4	1	
	5	1	5	10	10	5	1
	6	1	6	15	20	15	6
	7	1	7	21	35	35	21
	8	1	8	28	56	70	56
	9	1	9	36	84	126	126
	10	1	10	45	120	210	252
	11	1	11	55	165	330	462
	12	1	12	66	220	495	792

Computing $C(n, k)$: pseudocode and analysis

ALGORITHM *Binomial*(n, k)

//Computes $C(n, k)$ by the dynamic programming algorithm

//Input: A pair of nonnegative integers $n \geq k \geq 0$

//Output: The value of $C(n, k)$

for $i \leftarrow 0$ **to** n **do**

for $j \leftarrow 0$ **to** $\min(i, k)$ **do**

if $j = 0$ **or** $j = i$

$C[i, j] \leftarrow 1$

else $C[i, j] \leftarrow C[i - 1, j - 1] + C[i - 1, j]$

return $C[n, k]$

Time efficiency: $\Theta(nk)$

Space efficiency: $\Theta(nk)$

Longest Common Subsequence (LCS) Problem

LCS Problem: Overview

- The LCS problem is to find the longest subsequence common to all sequences in a set of sequences (often just two).
- Note that a subsequence is different from a substring in the sense that a subsequence need not be consecutive terms of the original sequence.
- An algorithm for the LCS problem could be used to find the longest common subsequence between the DNA strands of two organisms.
- For a given length of the two DNA strands, the longer the common subsequence, the more similar and closer (evolutionarily) are the two organisms.
- Example: $X = \text{ATGCAC}$ $Y = \text{CAGATCCA}$
 - $\text{LCS}(X, Y) = \text{ATCA}$.

LCS Problem: Idea

- Let the two sequences to compare be X of length m and Y of length n. We want to find the $LCS(X[1\dots m], Y[1\dots n])$.
- If $X[m] = Y[n]$, then we can simply discard the last character (that is common) from both the sequences and find the LCS of $X[1\dots m-1]$ and $Y[1\dots n-1]$, such that

$$LCS(X[1\dots m], Y[1\dots n]) = LCS(X[1\dots m-1], Y[1\dots n-1]) + 1.$$

- If $X[m] \neq Y[n]$, then the longest common subsequence of the two sequences can be at most either $X[m]$ or $Y[n]$; but not both. Hence, we can say that:

$$LCS(X[1\dots m], Y[1\dots n]) \\ = \text{Max} \{LCS(X[1\dots m-1], Y[1\dots n]), LCS(X[1\dots m], Y[1\dots n-1])\}$$

Dynamic Programming Formulation

Define: $LCS[i][j]$ = Length of the LCS of sequence $X[1\dots i]$ and $Y[1\dots j]$

Thus, $LCS[i][0] = 0$ for all i

$LCS[0][j] = 0$ for all j

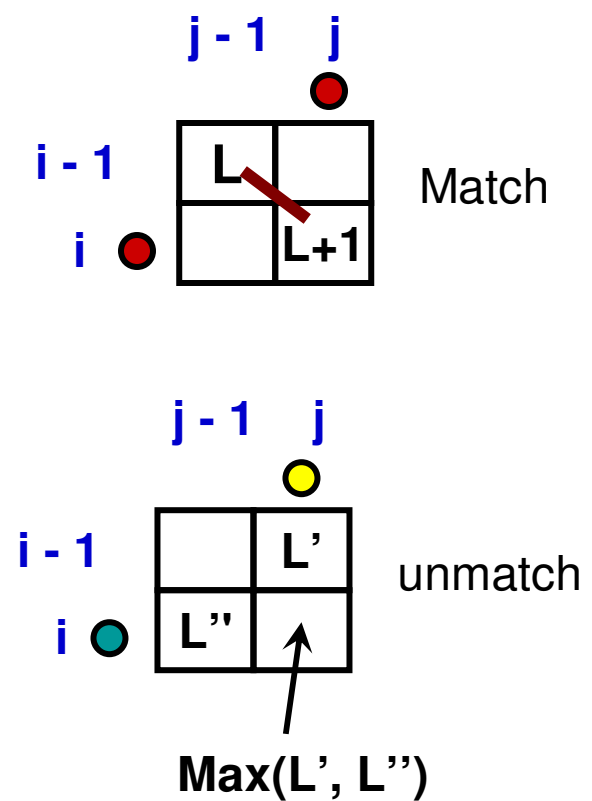
The goal is to find $LCS[m][n]$

$$LCS[i][j] = \begin{cases} LCS[i-1][j-1] + 1 & X[i] = Y[j] \\ \text{Max}\{LCS[i][j-1], LCS[i-1][j]\} & X[i] \neq Y[j] \end{cases}$$

X = ATGACTATAA
 Y = GACTAATA

LCS Example 1 (1)

		G	A	C	T	A	A	T	A
A	0	0	1	1	1	1	1	1	1
T	0	0	0	1	2	2	2	2	2
G	0	1	1	1	2	2	2	2	2
A	0	1	2	2	2	3	3	3	3
C	0	1	2	3	3	3	3	3	3
T	0	1	2	3	4	4	4	4	4
A	0	1	2	3	4	5	5	5	5
T	0	1	2	3	4	5	5	6	6
A	0	1	2	3	4	5	6	6	7
A	0	1	2	3	4	5	6	6	7



X = ATGACTATAA
 Y = GACTAATA

LCS Example 1 (2)

		G	A	C	T	A	A	T	A
A	0	0	0	0	0	0	0	0	0
T	0	0	0	1	2	2	2	2	2
G	0	1	1	1	2	2	2	2	2
A	0	1	2	2	2	3	3	3	3
C	0	1	2	3	3	3	3	3	3
T	0	1	2	3	4	4	4	4	4
A	0	1	2	3	4	5	5	5	5
T	0	1	2	3	4	5	5	6	6
A	0	1	2	3	4	5	6	6	7
A	0	1	2	3	4	5	6	6	7

Sequence Alignment

```

A T G A C T - A T A A
- - G A C T A A T - A
  
```

LCS: G A C T A T A

Ties are broken by going up

When you skip a symbol, it is aligned with a blank -.

X = TGACTAC
Y = ACTGATGC

LCS Example 2 (1)

		T	G	A	C	T	A	C
		0	0	0	0	0	0	0
A		0	0	0	1	1	1	1
C		0	0	0	1	2	2	2
T		0	1	1	1	2	3	3
G		0	1	2	2	2	3	3
A		0	1	2	3	3	3	4
T		0	1	2	3	3	4	4
G		0	1	2	3	3	4	4
C		0	1	2	3	4	4	4
		0	1	2	3	4	4	5

X = TGACTAC
 Y = ACTGATGC

LCS Example 2 (2)

	T	G	A	C	T	A	C
A	0	0	0	1	1	1	1
C	0	0	1	2	2	2	2
T	0	1	1	2	3	3	3
G	0	1	2	2	3	3	3
A	0	1	2	3	3	4	4
T	0	1	2	3	4	4	4
G	0	1	2	3	4	4	4
C	0	1	2	3	4	4	5

T G A C T - A - - C
 - - A C T G A T G C

LCS: A C T A C

LCS Example 3 (1)

	C	A	A	G	T	A	C	G
A	0	0	0	0	0	0	0	0
C	0	1	1	1	1	1	2	2
T	0	1	1	1	2	2	2	2
G	0	1	1	1	2	2	2	3
G	0	1	1	1	2	2	2	3
A	0	1	2	2	2	3	3	3
G	0	1	2	2	3	3	3	4
C	0	1	2	2	3	3	4	4
A	0	1	2	3	3	4	4	4
T	0	1	2	3	3	4	4	4

X = CAAGTACG
Y = ACTGGAGCAT

LCS Example 3 (2)

	C	A	A	G	T	A	C	G
A	0	0	0	0	0	0	0	0
C	0	1	1	1	1	1	2	2
T	0	1	1	1	2	2	2	2
G	0	1	1	1	2	2	2	3
G	0	1	1	1	2	2	2	3
A	0	1	2	2	2	3	3	3
C	0	1	2	2	3	3	3	4
A	0	1	2	3	3	3	4	4
T	0	1	2	3	3	4	4	4

X = CAAGTACG
Y = ACTGGAGCAT

C A A G - T - - A C G - - -
- - A - C T G G A - G C A T

LCS: A T A G

Coin Change Problem

- Given a set of coin denominations $CD[1 \dots N]$ and a value S , we want to determine the optimal (minimum) number of coins that can be used so that the coin values add up to S .
- Assume there is an infinite supply of the coins for each value.
- Unlike the greedy approach, the dynamic programming solution will work for all coin denominations.
- Example: $CD[1 \dots 3] = \{3, 1, 4\}$; $S = 6$
 - Greedy approach will give a solution of picking 3 coins (values: 4, 1, 1)
 - Dynamic programming approach will give a solution of picking 2 coins (values: 3, 3)

Coin Change Problem

Recurrence Relation

- Given S and $CD[1 \dots N]$
- Let $MNC^j[V]$ be the minimum number of coins that need to be picked up by considering coins at index $1 \dots j$ so that the coins picked up add to a value of V , where $0 \leq V \leq S$.
- Let $LCP^j[V] = CD[j]$ if the j th coin needs to be picked up so that the total value of the coins picked is V .

LCP – Last Coin value Picked

$$MNC^j[V] = \text{Minimum} \begin{cases} MNC^{j-1}[V] & \text{// if coin index } j \text{ should not} \\ & \text{// be picked up for an optimal} \\ & \text{// solution to add up to } V \\ 1 + MNC^j[V - CD[j]] & \text{// if picking the coin at index } j \text{ will reduce the} \\ & \text{// number of coins from what is known currently} \end{cases}$$

$0 \leq V \leq S$
 $1 \leq j \leq N$
 $V \geq CD[j]$

$MNC^j[X] = \infty$ for $X > 0$ and $j = 0$ (i.e., no coin is available for pickup)

$MNC^j[X] = 0$ for $X = 0$ and any j (i.e., value of the coins to add up to is 0)

Time complexity: $\Theta(N \cdot S)$;

Space Complexity: $\Theta(S)$

Iteration j-1

$v \rightarrow$	0	1	2	3	4	5	6	S	
MNC	0	Y
LCP	-	Z

Iteration j

Consider a value V (say, $V = 5$) that is greater than or equal to $CD[j]$ (say, $CD[j] = 3$).

Look $CD[j] = 3$ cells backwards (i.e., $V - CD[j]$)



$v \rightarrow$	0	1	2	3	4	5	6	S	
MNC	0	...	P	R
LCP	-	Q

$$R = \text{Min} (Y , 1 + P)$$

$$Q = Z \text{ if } R = Y$$

$$Q = CD[j] \text{ if } R = 1 + P$$

CD Array

j CD[j]
1 3
2 1
3 4

Coin Change Problem

Example 1 (Initialization)

MNC Table

Coins/Denomination		Values, V						
j	CD[j]	0	1	2	3	4	5	6
0	-	0	∞	∞	∞	∞	∞	∞
1	3	0	[Yellow shaded area]					
2	1	0						
3	4	0						

LCP Table

Coins/Denomination		Values, V						
j	CD[j]	0	1	2	3	4	5	6
0	-	-	-	-	-	-	-	-
1	3	-	[Yellow shaded area]					
2	1	-						
3	4	-						

CD Array

j CD[j]
1 3
2 1
3 4

Coin Change Problem

Example 1 (Iteration 1)

MNC Table

Coins/Denomination		Values, V						
j	CD[j]	0	1	2	3	4	5	6
0	-	0	∞	∞	∞	∞	∞	∞
1	3	0	∞	∞	1	∞	∞	2
2	1	0						
3	4	0						

LCP Table

Coins/Denomination		Values, V						
j	CD[j]	0	1	2	3	4	5	6
0	-	-	-	-	-	-	-	-
1	3	-	-	-	3	-	-	3
2	1	-						
3	4	-						

CD Array

j CD[j]
1 3
2 1
3 4

Coin Change Problem

Example 1 (Iteration 2)

MNC Table

Coins/Denomination		Values, V						
j	CD[j]	0	1	2	3	4	5	6
0	-	0						
1	3	0	∞	∞	1	∞	∞	2
2	1	0	1	2	1	2	3	2
3	4	0						

LCP Table

Coins/Denomination		Values, V						
j	CD[j]	0	1	2	3	4	5	6
0	-	-						
1	3	-	-	-	3	-	-	3
2	1	-	1	1	3	1	1	3
3	4	-						

CD Array

j CD[j]
1 3
2 1
3 4

Coin Change Problem

Example 1 (Iteration 3)

MNC Table

Coins/Denomination		Values, V						
j	CD[j]	0	1	2	3	4	5	6
0	-	0						
1	3	0						
2	1	0	1	2	1	2	3	2
3	4	0	1	2	1	1	2	2

LCP Table

Coins/Denomination		Values, V						
j	CD[j]	0	1	2	3	4	5	6
0	-	-						
1	3	-						
2	1	-	1	1	3	1	1	3
3	4	-	1	1	3	4	4	3

Coin Change Problem

Example 1 (Final Tables)

CD Array

j CD[j]

1 3

2 1

3 4

MNC Table

Coins/Denomination		Values, V						
j	CD[j]	0	1	2	3	4	5	6
0	-	0	∞	∞	∞	∞	∞	∞
1	3	0	∞	∞	1	∞	∞	2
2	1	0	1	2	1	2	3	2
3	4	0	1	2	1	1	2	2

LCP Table

Coins/Denomination		Values, V						
j	CD[j]	0	1	2	3	4	5	6
0	-	-	-	-	-	-	-	-
1	3	-	-	-	3	-	-	3
2	1	-	1	1	3	1	1	3
3	4	-	1	1	3	4	4	3

Tracing the solution (V = 6): Coins picked: 3, 3

MNC[6] = 2 coins to be picked up for Value = 6

LCP[6] = 3; So, pick coin of value 3 and go to LCP[6-3] = LCP[3]

LCP[3] = 3; So, pick coin of value 3 and go to LCP[3-3] = LCP[0] = - // Done

Coin Change Problem

Example 2

CD Array

j 1 2 3 4
 CD[j] 5 1 2 4

Let S = 17

Initialization

MNC

Coins/ Denomination		Values, V																	
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	-	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
1	5	0																	
2	1	0																	
3	2	0																	
4	4	0																	

LCP

Coins/ Denomination		Values, V																	
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
1	5	-																	
2	1	-																	
3	2	-																	
4	4	-																	

Coin Change Problem

Example 2

CD Array

j 1 2 3 4
 CD[j] 5 1 2 4

Iteration 1

Let S = 17

MNC

Coins/ Denomination		Values, V																	
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	-	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
1	5	0	∞	∞	∞	∞	1	∞	∞	∞	∞	2	∞	∞	∞	∞	3	∞	∞
2	1	0																	
3	2	0																	
4	4	0																	

LCP

Coins/ Denomination		Values, V																	
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
1	5	-	-	-	-	-	5	-	-	-	-	5	-	-	-	-	5	-	-
2	1	-																	
3	2	-																	
4	4	-																	

Coin Change Problem

Example 2

CD Array

j 1 2 3 4
 CD[j] 5 1 2 4

Iteration 2

Let S = 17

MNC

Coins/ Denomination		Values, V																		
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	
0	-	0																		
1	5	0	∞	∞	∞	∞	1	∞	∞	∞	∞	2	∞	∞	∞	∞	3	∞	∞	
2	1	0	1	2	3	4	1	2	3	4	5	2	3	4	5	6	3	4	5	
3	2	0																		
4	4	0																		

LCP

Coins/ Denomination		Values, V																		
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	
0	-	-																		
1	5	-	-	-	-	-	5	-	-	-	-	5	-	-	-	-	5	-	-	
2	1	-	1	1	1	1	5	1	1	1	1	5	1	1	1	1	5	1	1	
3	2	-																		
4	4	-																		

Coin Change Problem

Example 2

CD Array

j 1 2 3 4
 CD[j] 5 1 2 4

Iteration 3

Let S = 17

MNC

Coins/ Denomination		Values, V																	
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	-	0																	
1	5	0																	
2	1	0	1	2	3	4	1	2	3	4	5	2	3	4	5	6	3	4	5
3	2	0	1	1	2	3	1	2	2	3	4	2	3	3	4	5	3	4	4
4	4	0																	

LCP

Coins/ Denomination		Values, V																	
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	-	-																	
1	5	-																	
2	1	-	1	1	1	1	5	1	1	1	1	5	1	1	1	1	5	1	1
3	2	-	1	2	2	2	5	1	2	2	2	5	1	2	2	2	5	1	2
4	4	-																	

Coin Change Problem

Example 2

CD Array

j 1 2 3 4
 CD[j] 5 1 2 4

Iteration 4

Let S = 17

MNC

Coins/ Denomination		Values, V																	
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	-	0																	
1	5	0																	
2	1	0																	
3	2	0	1	1	2	3	1	2	2	3	4	2	3	3	4	5	3	4	4
4	4	0	1	1	2	1	1	2	2	2	2	2	3	3	3	3	3	4	4

LCP

Coins/ Denomination		Values, V																	
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	-	-																	
1	5	-																	
2	1	-																	
3	2	-	1	2	2	2	5	1	2	2	2	5	1	2	2	2	5	1	2
4	4	-	1	2	2	4	5	1	2	4	4	5	1	2	4	4	5	1	2

Coin Change Problem

Example 2

CD Array

j	1	2	3	4
CD[j]	5	1	2	4

Let S = 17

Final Tables

MNC

Coins/ Denomination		Values, V																	
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	-	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
1	5	0	∞	∞	∞	∞	1	∞	∞	∞	∞	2	∞	∞	∞	∞	3	∞	∞
2	1	0	1	2	3	4	1	2	3	4	5	2	3	4	5	6	3	4	5
3	2	0	1	1	2	3	1	2	2	3	4	2	3	3	4	5	3	4	4
4	4	0	1	1	2	1	1	2	2	2	2	2	3	3	3	3	3	4	4

LCP

Coins/ Denomination		Values, V																	
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
1	5	-	-	-	-	-	5	-	-	-	-	5	-	-	-	-	5	-	-
2	1	-	1	1	1	1	5	1	1	1	1	5	1	1	1	1	5	1	1
3	2	-	1	2	2	2	5	1	2	2	2	5	1	2	2	2	5	1	2
4	4	-	1	2	2	4	5	1	2	4	4	5	1	2	4	4	5	1	2

CD Array

j	1	2	3	4
CD[j]	5	1	2	4

Final Table

V →	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
MNC	0	1	1	2	1	1	2	2	2	2	2	3	3	3	3	3	4	4
LCP	-	1	2	2	4	5	1	2	4	4	5	1	2	4	4	5	1	2

Tracing the solution (V = 17):

MNC[17] = 4 coins to be picked up for Value = 17

LCP[17] = 2; So, pick coin of value 2 and go to LCP[17 - 2] = LCP[15]

LCP[13] = 5; So, pick coin of value 5 and go to LCP[15 - 5] = LCP[10]

LCP[10] = 5; So, pick coin of value 5 and go to LCP[10 - 5] = LCP[5]

LCP[5] = 5; So, pick coin of value 5 and go to LCP[5 - 5] = LCP[0] = - // Done!!

Coins picked for Value = 17 are: 2, 5, 5, 5

Tracing the solution (V = 12):

MNC[12] = 3 coins to be picked up for Value = 12

LCP[12] = 2; So, pick coin of value 2 and go to LCP[12 - 2] = LCP[10]

LCP[10] = 5; So, pick coin of value 5 and go to LCP[10 - 5] = LCP[5]

LCP[5] = 5; So, pick coin of value 5 and go to LCP[5 - 5] = LCP[0] = - // Done!!

Coins picked for Value = 12 are: 2, 5, 5

Coin Change Problem

Example 3

CD Array

j 1 2 3 4
 CD[j] 5 6 1 9

Let S = 11

Initialization

MNC

Coins/ Denomination		Values, V												
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11	
0	-	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	
1	5	0												
2	6	0												
3	1	0												
4	9	0												

LCP

Coins/ Denomination		Values, V												
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11	
0	-	-	-	-	-	-	-	-	-	-	-	-	-	
1	5	-												
2	6	-												
3	1	-												
4	9	-												

Coin Change Problem

Example 3

CD Array

j 1 2 3 4
 CD[j] 5 6 1 9

Let S = 11

Iteration 1

MNC

Coins/ Denomination		Values, V											
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11
0	-	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
1	5	0	∞	∞	∞	∞	1	∞	∞	∞	∞	2	∞
2	6	0											
3	1	0											
4	9	0											

LCP

Coins/ Denomination		Values, V											
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11
0	-	-	-	-	-	-	-	-	-	-	-	-	-
1	5	-	-	-	-	-	5	-	-	-	-	5	-
2	6	-											
3	1	-											
4	9	-											

Coin Change Problem

Example 3

CD Array

j 1 2 3 4
 CD[j] 5 6 1 9

Let S = 11

Iteration 2

MNC

Coins/ Denomination		Values, V												
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11	
0	-	0												
1	5	0	∞	∞	∞	∞	1	∞	∞	∞	∞	2	∞	
2	6	0	∞	∞	∞	∞	1	1	∞	∞	∞	2	2	
3	1	0												
4	9	0												

LCP

Coins/ Denomination		Values, V												
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11	
0	-	-												
1	5	-	-	-	-	-	5	-	-	-	-	5	-	
2	6	-	-	-	-	-	5	6	-	-	-	5	6	
3	1	-												
4	9	-												

Coin Change Problem

Example 3

CD Array

j 1 2 3 4
 CD[j] 5 6 1 9

Let S = 11

Iteration 3

MNC

Coins/ Denomination		Values, V											
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11
0	-	0											
1	5	0											
2	6	0	∞	∞	∞	∞	1	1	∞	∞	∞	2	2
3	1	0	1	2	3	4	1	1	2	3	4	2	2
4	9	0											

LCP

Coins/ Denomination		Values, V											
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11
0	-	-											
1	5	-											
2	6	-	-	-	-	-	5	6	-	-	-	5	6
3	1	-	1	1	1	1	5	6	1	1	1	5	6
4	9	-											

Coin Change Problem

Example 3

CD Array

j 1 2 3 4
 CD[j] 5 6 1 9

Let S = 11

Iteration 4

MNC

Coins/ Denomination		Values, V											
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11
0	-	0											
1	5	0											
2	6	0											
3	1	0	1	2	3	4	1	1	2	3	4	2	2
4	9	0	1	2	3	4	1	1	2	3	1	2	2

LCP

Coins/ Denomination		Values, V											
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11
0	-	-											
1	5	-											
2	6	-											
3	1	-	1	1	1	1	5	6	1	1	1	5	6
4	9	-	1	1	1	1	5	6	1	1	9	5	6

MNC

Coins/ Denomination		Values, V											
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11
0	-	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
1	5	0	∞	∞	∞	∞	1	∞	∞	∞	∞	2	∞
2	6	0	∞	∞	∞	∞	1	1	∞	∞	∞	2	2
3	1	0	1	2	3	4	1	1	2	3	4	2	2
4	9	0	1	2	3	4	1	1	2	3	1	2	2

CD Array

j 1 2 3 4
 CD[j] 5 6 1 9

Let S = 11

LCP

Coins/ Denomination		Values, V											
j	CD[j]	0	1	2	3	4	5	6	7	8	9	10	11
0	-	-	-	-	-	-	-	-	-	-	-	-	-
1	5	-	-	-	-	-	5	-	-	-	-	5	-
2	6	-	-	-	-	-	5	6	-	-	-	5	6
3	1	-	1	1	1	1	5	6	1	1	1	5	6
4	9	-	1	1	1	1	5	6	1	1	9	5	6

Final Tables

Tracing the Solution for V = 11

LCP[11] = 6; Pick 6
 LCP[11 - 6] = LCP[5]
 LCP[5] = 5; Pick 5
 LCP[5 - 5] = - Done!!

Coins Picked for S = 11 are: 6, 5 (2 coins – optimal)

Greedy approach would have given a solution of 3 coins (9, 1, 1)

Integer Knapsack Problem

- **Problem Statement**: Design a dynamic programming algorithm for the integer-knapsack problem: given n items of known weights w_1, w_2, \dots, w_n (where all the weights are integers) and values v_1, v_2, \dots, v_n (the values need not be integers), and a knapsack capacity W (an integer), find the most valuable subset of the items that fit into the knapsack.
- **Solution**: Let $F(i, j)$ be the value of the most valuable subset of the first i items ($1 \leq i \leq n$) that fit into the knapsack of capacity j ($1 \leq j \leq W$). We can divide all the subsets of the first i items that fit into the knapsack of capacity j into two categories: those that do not include the i^{th} item and those that do.
 - Among the subsets that do not include the i^{th} item, the value of an optimal subset is $F(i-1, j)$.
 - Among the subsets that do include the i^{th} item (hence, $j - w_i \geq 0$), an optimal subset is made up of this item and an optimal subset of the first $i-1$ items that fits into the knapsack of capacity $j - w_i$. The value of such an optimal subset is $v_i + F(i-1, j - w_i)$.

$$F(i, j) = \begin{cases} \max\{F(i-1, j), v_i + F(i-1, j - w_i)\} & \text{if } j - w_i \geq 0, \\ F(i-1, j) & \text{if } j - w_i < 0. \end{cases}$$

Initial Condition: $F(0, j) = 0$ for $1 \leq j \leq W$ $F(i, 0) = 0$ for $1 \leq i \leq n$

Idea to Solve the Int. Knapsack Prob.

- The goal is to find $F(n, W)$, the optimal value of a subset of the n given items that fit into the knapsack of capacity W , and an optimal subset itself.
- For $i, j > 0$, to compute the entry in the i^{th} row and j^{th} column, $F(i, j)$, we compute the maximum of the entry in the previous row and the same column and the sum of v_i and the entry in the previous row and w_i columns to the left.
- The table can be filled either row-wise or column-wise.

		0	$j - w_i$	j	W
	0	0	0	0	0
	$i - 1$	0	$F(i - 1, j - w_i)$	$F(i - 1, j)$	
w_i, v_i	i	0		$F(i, j)$	
	n	0			goal

Example 1: Integer Knapsack Problem

- Find the composition of items that maximizes the value of the knapsack of integer capacity-weight 5.

item	weight	value
1	2	\$12
2	1	\$10
3	3	\$20
4	2	\$15

		Capacity, j					
		0	1	2	3	4	5
	i	0	0	0	0	0	0
w1 = 2, v1 = 12	1	0					
w2 = 1, v2 = 10	2	0					
w3 = 3, v3 = 20	3	0					
w4 = 2, v4 = 15	4	0					

		Capacity, j					
		0	1	2	3	4	5
	i	0	0	0	0	0	0
w1 = 2, v1 = 12	1	0					
w2 = 1, v2 = 10	2	0					
w3 = 3, v3 = 20	3	0					
w4 = 2, v4 = 15	4	0					

Example 1: Integer Knapsack Problem

- Find the composition of items that maximizes the value of the knapsack of integer capacity-weight 5.

item	weight	value
1	2	\$12
2	1	\$10
3	3	\$20
4	2	\$15

		Capacity, j						
		i	0	1	2	3	4	5
	0	0	0	0	0	0	0	0
w1 = 2, v1 = 12	1	0	0	0	12	12	12	12
w2 = 1, v2 = 10	2	0						
w3 = 3, v3 = 20	3	0						
w4 = 2, v4 = 15	4	0						

		Capacity, j						
		i	0	1	2	3	4	5
	0	0	0	0	0	0	0	0
w1 = 2, v1 = 12	1	0	0	$C[0,0]+w1$	$C[0,1]+w1$	$C[0,2]+w1$	$C[0,3]+w1$	
w2 = 1, v2 = 10	2	0						
w3 = 3, v3 = 20	3	0						
w4 = 2, v4 = 15	4	0						

Example 1: Integer Knapsack Problem

- Find the composition of items that maximizes the value of the knapsack of integer capacity-weight 5.

item	weight	value
1	2	\$12
2	1	\$10
3	3	\$20
4	2	\$15

		Capacity, j					
		0	1	2	3	4	5
i		0					
w1 = 2, v1 = 12	1	0	0	12	12	12	12
w2 = 1, v2 = 10	2	0	10	12	22	22	22
w3 = 3, v3 = 20	3	0					
w4 = 2, v4 = 15	4	0					

		Capacity, j					
		0	1	2	3	4	5
i		0					
w1 = 2, v1 = 12	1	0	0	$C[0,0]+w1$	$C[0,1]+w1$	$C[0,2]+w1$	$C[0,3]+w1$
w2 = 1, v2 = 10	2	0	$C[1,0]+w2$	$C[1,2]$	$C[1,2]+w2$	$C[1,3]+w2$	$C[1,4]+w2$
w3 = 3, v3 = 20	3	0					
w4 = 2, v4 = 15	4	0					

Example 1: Integer Knapsack Problem

- Find the composition of items that maximizes the value of the knapsack of integer capacity-weight 5.

item	weight	value
1	2	\$12
2	1	\$10
3	3	\$20
4	2	\$15

		Capacity, j					
		0	1	2	3	4	5
	i	0					
	0	0					
w1 = 2, v1 = 12	1	0					
w2 = 1, v2 = 10	2	0	10	12	22	22	22
w3 = 3, v3 = 20	3	0	10	12	22	30	32
w4 = 2, v4 = 15	4	0					

		Capacity, j					
		0	1	2	3	4	5
	i	0					
	0	0					
w1 = 2, v1 = 12	1	0					
w2 = 1, v2 = 10	2	0	$C[1,0]+w2$	$C[1,2]$	$C[1,2]+w2$	$C[1,3]+w2$	$C[1,4]+w2$
w3 = 3, v3 = 20	3	0	$C[2,1]$	$C[2,2]$	$C[2,3]$	$C[2,1]+w3$	$C[2,2]+w3$
w4 = 2, v4 = 15	4	0					

Example 1: Integer Knapsack Problem

- Find the composition of items that maximizes the value of the knapsack of integer capacity-weight 5.

item	weight	value
1	2	\$12
2	1	\$10
3	3	\$20
4	2	\$15

		Capacity, j					
		0	1	2	3	4	5
	i	0					
	0	0					
w1 = 2, v1 = 12	1	0					
w2 = 1, v2 = 10	2	0					
w3 = 3, v3 = 20	3	0	10	12	22	30	32
w4 = 2, v4 = 15	4	0	10	15	25	30	37

		Capacity, j					
		0	1	2	3	4	5
	i	0					
	0	0					
w1 = 2, v1 = 12	1	0					
w2 = 1, v2 = 10	2	0					
w3 = 3, v3 = 20	3	0	$C[2,1]$	$C[2,2]$	$C[2,3]$	$C[2,1]+w3$	$C[2,2]+w3$
w4 = 2, v4 = 15	4	0	$C[3,1]$	$C[3,0]+w4$	$C[3,1]+w4$	$C[3,4]$	$C[3,3]+w4$

Example 1: Integer Knapsack Problem

- Find the composition of items that maximizes the value of the knapsack of integer capacity-weight 5.

item	weight	value
1	2	\$12
2	1	\$10
3	3	\$20
4	2	\$15

		Capacity, j						
		i	0	1	2	3	4	5
	0	0	0	0	0	0	0	0
w1 = 2, v1 = 12	1	0	0	12	12	12	12	12
w2 = 1, v2 = 10	2	0	10	12	22	22	22	22
w3 = 3, v3 = 20	3	0	10	12	22	30	32	32
w4 = 2, v4 = 15	4	0	10	15	25	30	37	37

		Capacity, j						
		i	0	1	2	3	4	5
	0	0	0	0	0	0	0	0
w1 = 2, v1 = 12	1	0	0	$C[0,0]+w1$	$C[0,1]+w1$	$C[0,2]+w1$	$C[0,3]+w1$	
w2 = 1, v2 = 10	2	0	$C[1,0]+w2$	$C[1,2]$	$C[1,2]+w2$	$C[1,3]+w2$	$C[1,4]+w2$	
w3 = 3, v3 = 20	3	0	$C[2,1]$	$C[2,2]$	$C[2,3]$	$C[2,1]+w3$	$C[2,2]+w3$	
w4 = 2, v4 = 15	4	0	$C[3,1]$	$C[3,0]+w4$	$C[3,1]+w4$	$C[3,4]$	$C[3,3]+w4$	

Choose W4(2), W2(1), W1(2), with values totaling to 37 and capacity 5.

Example 2: Integer Knapsack Problem

- Find the composition of items that maximizes the value of the knapsack of integer capacity-weight 6.

item	weight	value
1	3	\$25
2	2	\$20
3	1	\$15
4	4	\$40
5	5	\$50

		Capacity, j						
	i	0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
w1 = 3, v1 = 25	1	0						
w2 = 2, v2 = 20	2	0						
w3 = 1, v3 = 15	3	0						
w4 = 4, v4 = 40	4	0						
w5 = 5, v5 = 50	5	0						

		Capacity, j						
	i	0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
w1 = 3, v1 = 25	1	0						
w2 = 2, v2 = 20	2	0						
w3 = 1, v3 = 15	3	0						
w4 = 4, v4 = 40	4	0						
w5 = 5, v5 = 50	5	0						

Example 2: Integer Knapsack Problem

- Find the composition of items that maximizes the value of the knapsack of integer capacity-weight 6.

item	weight	value
1	3	\$25
2	2	\$20
3	1	\$15
4	4	\$40
5	5	\$50

		Capacity, j							
		i	0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0	0
w1 = 3, v1 = 25	1	0	0	0	25	25	25	25	25
w2 = 2, v2 = 20	2	0							
w3 = 1, v3 = 15	3	0							
w4 = 4, v4 = 40	4	0							
w5 = 5, v5 = 50	5	0							

		Capacity, j							
		i	0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0	0
w1 = 3, v1 = 25	1	0	$C[0,1]$	$C[0,2]$	$C[0,0]+w1$	$C[0,1]+w1$	$C[0,2]+w1$	$C[0,3]+w1$	
w2 = 2, v2 = 20	2	0							
w3 = 1, v3 = 15	3	0							
w4 = 4, v4 = 40	4	0							
w5 = 5, v5 = 50	5	0							

Example 2: Integer Knapsack Problem

- Find the composition of items that maximizes the value of the knapsack of integer capacity-weight 6.

item	weight	value
1	3	\$25
2	2	\$20
3	1	\$15
4	4	\$40
5	5	\$50

		Capacity, j						
		0	1	2	3	4	5	6
	i	0						
	0	0						
w1 = 3, v1 = 25	1	0	0	0	25	25	25	25
w2 = 2, v2 = 20	2	0	0	20	25	25	45	45
w3 = 1, v3 = 15	3	0						
w4 = 4, v4 = 40	4	0						
w5 = 5, v5 = 50	5	0						

		Capacity, j						
		0	1	2	3	4	5	6
	i	0						
	0	0						
w1 = 3, v1 = 25	1	0	$C[0,1]$	$C[0,2]$	$C[0,0]+w1$	$C[0,1]+w1$	$C[0,2]+w1$	$C[0,3]+w1$
w2 = 2, v2 = 20	2	0	$C[1,1]$	$C[1,0]+w2$	$C[1,3]$	$C[1,4]$	$C[1,3]+w2$	$C[1,4]+w2$
w3 = 1, v3 = 15	3	0						
w4 = 4, v4 = 40	4	0						
w5 = 5, v5 = 50	5	0						

Example 2: Integer Knapsack Problem

- Find the composition of items that maximizes the value of the knapsack of integer capacity-weight 6.

item	weight	value
1	3	\$25
2	2	\$20
3	1	\$15
4	4	\$40
5	5	\$50

		Capacity, j							
		i	0	1	2	3	4	5	6
		0	0						
$w_1 = 3, v_1 = 25$	1	0							
$w_2 = 2, v_2 = 20$	2	0	0	20	25	25	45	45	
$w_3 = 1, v_3 = 15$	3	0	15	20	35	40	45	60	
$w_4 = 4, v_4 = 40$	4	0							
$w_5 = 5, v_5 = 50$	5	0							

		Capacity, j							
		i	0	1	2	3	4	5	6
		0	0						
$w_1 = 3, v_1 = 25$	1	0							
$w_2 = 2, v_2 = 20$	2	0	$C[1,1]$	$C[1,2]+w_2$	$C[1,3]$	$C[1,4]$	$C[1,3]+w_2$	$C[1,4]+w_2$	
$w_3 = 1, v_3 = 15$	3	0	$C[0,0]+w_3$	$C[2,2]$	$C[2,2]+w_3$	$C[2,3]+w_3$	$C[2,5]$	$C[2,5]+w_3$	
$w_4 = 4, v_4 = 40$	4	0							
$w_5 = 5, v_5 = 50$	5	0							

Example 2: Integer Knapsack Problem

- Find the composition of items that maximizes the value of the knapsack of integer capacity-weight 6.

item	weight	value
1	3	\$25
2	2	\$20
3	1	\$15
4	4	\$40
5	5	\$50

		Capacity, j						
	i	0	1	2	3	4	5	6
	0	0						
w1 = 3, v1 = 25	1	0						
w2 = 2, v2 = 20	2	0						
w3 = 1, v3 = 15	3	0	15	20	35	40	45	60
w4 = 4, v4 = 40	4	0	15	20	35	40	55	60
w5 = 5, v5 = 50	5	0						

		Capacity, j						
	i	0	1	2	3	4	5	6
	0	0						
w1 = 3, v1 = 25	1	0						
w2 = 2, v2 = 20	2	0						
w3 = 1, v3 = 15	3	0	$C[0,0]+w3$	$C[2,2]$	$C[2,2]+w3$	$C[2,3]+w3$	$C[2,5]$	$C[2,5]+w3$
w4 = 4, v4 = 40	4	0	$C[3,1]$	$C[3,2]$	$C[3,3]$	$C[3,4]$	$C[3,1]+w4$	$C[3,6]$
w5 = 5, v5 = 50	5	0						

Example 2: Integer Knapsack Problem

- Find the composition of items that maximizes the value of the knapsack of integer capacity-weight 6.

item	weight	value
1	3	\$25
2	2	\$20
3	1	\$15
4	4	\$40
5	5	\$50

		Capacity, j							
		i	0	1	2	3	4	5	6
	0	0							
w1 = 3, v1 = 25	1	0							
w2 = 2, v2 = 20	2	0							
w3 = 1, v3 = 15	3	0							
w4 = 4, v4 = 40	4	0	15	20	35	40	55	60	
w5 = 5, v5 = 50	5	0	15	20	35	40	55	65	

		Capacity, j							
		i	0	1	2	3	4	5	6
	0	0							
w1 = 3, v1 = 25	1	0							
w2 = 2, v2 = 20	2	0							
w3 = 1, v3 = 15	3	0							
w4 = 4, v4 = 40	4	0	C[3,1]	C[3,2]	C[3,3]	C[3,4]	C[3,1]+w4	C[3,6]	
w5 = 5, v5 = 50	5	0	C[4,1]	C[4,2]	C[4,3]	C[4,4]	C[4,5]	C[4,1]+w5	

Example 2: Integer Knapsack Problem

- Find the composition of items that maximizes the value of the knapsack of integer capacity-weight 6.

item	weight	value
1	3	\$25
2	2	\$20
3	1	\$15
4	4	\$40
5	5	\$50

		Capacity, j							
		i	0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0	0
w1 = 3, v1 = 25	1	0	0	0	25	25	25	25	25
w2 = 2, v2 = 20	2	0	0	20	25	25	45	45	45
w3 = 1, v3 = 15	3	0	15	20	35	40	45	60	60
w4 = 4, v4 = 40	4	0	15	20	35	40	55	60	60
w5 = 5, v5 = 50	5	0	15	20	35	40	55	65	65

		Capacity, j							
		i	0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0	0
w1 = 3, v1 = 25	1	0	C[0,1]	C[0,2]	C[0,0]+w1	C[0,1]+w1	C[0,2]+w1	C[0,3]+w1	C[0,3]+w1
w2 = 2, v2 = 20	2	0	C[1,1]	C[1,0]+w2	C[1,3]	C[1,4]	C[1,3]+w2	C[1,4]+w2	C[1,4]+w2
w3 = 1, v3 = 15	3	0	C[0,0]+w3	C[2,2]	C[2,2]+w3	C[2,3]+w3	C[2,5]	C[2,5]+w3	C[2,5]+w3
w4 = 4, v4 = 40	4	0	C[3,1]	C[3,2]	C[3,3]	C[3,4]	C[3,1]+w4	C[3,6]	C[3,6]
w5 = 5, v5 = 50	5	0	C[4,1]	C[4,2]	C[4,3]	C[4,4]	C[4,5]	C[4,1]+w5	C[4,1]+w5

Choose W5(5) and W3(1) with values totaling to \$65 and capacity 6.

Matrix Multiplication

- Given two matrices $A_{m \times n}$ and $B_{n \times p}$, the requirement to be able to multiply $A \times B$ is that the number of columns (n) in the first matrix (A) should be the same as the number of rows (n) in the second matrix (B). The resulting product matrix is of dimension $m \times p$.
- $A_{m \times n} * B_{n \times p} = C_{m \times p}$
- We do 'n' multiplications to fill up a cell in the product matrix C. As there are $m \times p$ such cells, the total number of multiplications encountered in the above case is **$m \times n \times p$** .

- Example:

The diagram shows the multiplication of two matrices to produce a product matrix. The first matrix is a 3x4 matrix with values 3, 4, 1, 2 in the first row; 5, 2, 2, 1 in the second row; and 1, 3, 4, 2 in the third row. The second matrix is a 4x2 matrix with values 1, 2 in the first row; 4, 5 in the second row; 3, 2 in the third row; and 5, 1 in the fourth row. The product matrix is a 3x2 matrix with all cells containing 'x'. An arrow points from the top-left cell of the product matrix to the equation $3*1 + 4*4 + 1*3 + 2*5 = 32$. A double-headed arrow above the product matrix is labeled "4 multiplications / cell". A blue box on the right contains the text "Total number of Multiplications" and the equation $3 \times 4 \times 2 = 24$.

3	4	1	2
5	2	2	1
1	3	4	2

3x4

*

1	2
4	5
3	2
5	1

4x2

=

x	x
x	x
x	x

3x2

4 multiplications / cell

$3*1 + 4*4 + 1*3 + 2*5 = 32$

Total number of Multiplications
 $3 \times 4 \times 2 = 24$

Motivating Example

- Matrix multiplication is commutative.
- For example: $A*B*C$ can be multiplied as $(A*B)*C = A*(B*C)$
- Consider three matrices $A^1_{3 \times 4}$, $A^2_{4 \times 5}$, $A^3_{5 \times 2}$. How should we parenthesize them so that we do the minimum number of multiplications?

	# Multipl.
$A^1_{3 \times 4} * A^2_{4 \times 5} * A^3_{5 \times 2}$	
$= A^1_{3 \times 4} * (A^2_{4 \times 5} * A^3_{5 \times 2}) = A^1_{3 \times 4} * ()_{4 \times 2}$ <div style="display: flex; justify-content: space-around; margin-top: 5px;"> $4 \times 5 \times 2 = 40$ $3 \times 4 \times 2 = 24$ </div>	$40 + 24$ $= 64$
$= (A^1_{3 \times 4} * A^2_{4 \times 5}) * A^3_{5 \times 2} = ()_{3 \times 5} * A^3_{5 \times 2}$ <div style="display: flex; justify-content: space-around; margin-top: 5px;"> $3 \times 4 \times 5 = 60$ $3 \times 5 \times 2 = 30$ </div>	$60 + 30$ $= 90$

So, the best way to parenthesize is **$A1 * (A2 * A3)$**

Matrix Chain Multiplication

- Given a sequence of matrices, A_1, \dots, A_n : the problem is to find the best way to parenthesize them so that we do the minimum number of multiplications.

$$\begin{array}{cccccccc}
 A_1 & \times & A_2 & \times & A_3 & \times & \dots & \times & A_n \\
 p_0 \times p_1 & & p_1 \times p_2 & & p_2 \times p_3 & & \dots & & p_{n-1} \times p_n
 \end{array}$$

Optimal Substructure

Given a spread of matrices $A_i \times \dots \times A_j$, we need to find a 'k' such that $i \leq k < j$ such that multiplying $A_i \times \dots \times A_j = (A_i \times \dots \times A_k) * (A_{k+1} \times \dots \times A_j)$ would result in the minimum number of multiplications for all possible values of k; $i \leq k < j$

Let $M[i, j]$ indicate the minimum number of multiplications needed to compute $A_i \times \dots \times A_j$, where $i \leq j$

$$M[i, j] = \begin{cases} 0 & \text{if } i = j \\ \text{Min}_{i \leq k < j} & \left(M[i, k] + M[k+1, j] + p_{i-1} * p_k * p_j \right) \end{cases}$$

Example to Illustrate the Recurrence

$$M[i, j] = \begin{cases} 0 & \text{if } i = j \\ \text{Min}_{i \leq k < j} \left(M[i, k] + M[k+1, j] + p_{i-1} * p_k * p_j \right) \end{cases}$$

The dimension of the product matrix $(A_i \times \dots \times A_k)$ is $p_{i-1} \times p_k$

$$p_{i-1} \times p_i \quad p_{k-1} \times p_k$$

The dimension of the product matrix $(A_{k+1} \times \dots \times A_j)$ is $p_k \times p_j$

$$p_k \times p_{k+1} \quad p_{j-1} \times p_j$$

Let us say, we want to find $A_2 \dots A_7$ and we decide to try for $k = 4$

That is, we want to multiply

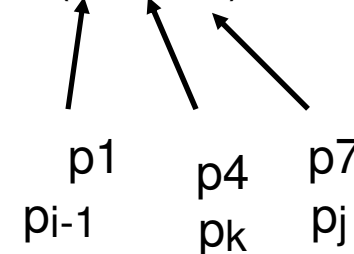
$$A_2 \times \dots \times A_7 = (A_2 \times \dots \times A_4) * (A_5 \times \dots \times A_7)$$

Dimensions: 4×3 4×7 7×3

A1	$p_0 \times p_1$	3x4
A2	$p_1 \times p_2$	4x5
A3	$p_2 \times p_3$	5x2
A4	$p_3 \times p_4$	2x7
A5	$p_4 \times p_5$	7x3
A6	$p_5 \times p_6$	3x5
A7	$p_6 \times p_7$	5x3
A8	$p_7 \times p_8$	3x6

$M[2, 7]$ for $k = 4$ is

$$M[2, 4] + M[5, 7] + (4 * 7 * 3)$$



EXAMPLE 1

A1	p0 x p1	3x4
A2	p1 x p2	4x5
A3	p2 x p3	5x2
A4	p3 x p4	2x7
A5	p4 x p5	7x2

$$M[i, j] = \begin{cases} 0 & \text{if } i = j \\ \text{Min}_{i \leq k < j} & \left(M[i, k] + M[k+1, j] + p_{i-1} * p_k * p_j \right) \end{cases}$$

Initialization

M-table

	j = 1	j = 2	j = 3	j = 4	j = 5
i = 1	0				
i = 2		0			
i = 3			0		
i = 4				0	
i = 5					0

k-table

	j = 1	j = 2	j = 3	j = 4	j = 5
i = 1	-				
i = 2		-			
i = 3			-		
i = 4				-	
i = 5					-

EXAMPLE 1

A1	p0 x p1	3x4
A2	p1 x p2	4x5
A3	p2 x p3	5x2
A4	p3 x p4	2x7
A5	p4 x p5	7x2

$$M[i, j] = \begin{cases} 0 & \text{if } i = j \\ \text{Min}_{i \leq k < j} & \left(M[i, k] + M[k+1, j] + p_{i-1} * p_k * p_j \right) \end{cases}$$

Note: Spread = j - i

Spread of '1'

M-table

	j = 1	j = 2	j = 3	j = 4	j = 5
i = 1	0	60			
i = 2		0	40		
i = 3			0	70	
i = 4				0	28
i = 5					0

k-table

	j = 1	j = 2	j = 3	j = 4	j = 5
i = 1	-	1			
i = 2		-	2		
i = 3			-	3	
i = 4				-	4
i = 5					-

p0	p1	p2	p3	p4	p5
3	4	5	2	7	2

$$M[i, j] = M[i, 1] + M[2, 2] + p_0 * p_1 * p_2 \quad \{k=1\}$$

$$= 0 + 0 + 3 * 4 * 5 = 60$$

$$M[i, j] = M[2, 2] + M[3, 3] + p_1 * p_2 * p_3 \quad \{k=2\}$$

$$= 0 + 0 + 4 * 5 * 2 = 40$$

$$M[i, j] = M[3, 3] + M[4, 4] + p_2 * p_3 * p_4 \quad \{k=3\}$$

$$= 0 + 0 + 5 * 2 * 7 = 70$$

$$M[i, j] = M[4, 4] + M[5, 5] + p_3 * p_4 * p_5 \quad \{k=4\}$$

$$= 0 + 0 + 2 * 7 * 2 = 28$$

$$M[i, j] = \begin{cases} 0 & \text{if } i = j \\ \text{Min}_{i \leq k < j} & \left(M[i, k] + M[k+1, j] + p_{i-1} * p_k * p_j \right) \end{cases}$$

EXAMPLE 1

A1	p0 x p1	3x4
A2	p1 x p2	4x5
A3	p2 x p3	5x2
A4	p3 x p4	2x7
A5	p4 x p5	7x2

$$M[i, j] = \begin{cases} 0 & \text{if } i = j \\ \text{Min}_{i \leq k < j} & \left(M[i, k] + M[k+1, j] + p_{i-1} * p_k * p_j \right) \end{cases}$$

Spread of '2'

M-table

	j = 1	j = 2	j = 3	j = 4	j = 5
i = 1	0	60	64		
i = 2		0	40	96	
i = 3			0	70	48
i = 4				0	28
i = 5					0

k-table

	j = 1	j = 2	j = 3	j = 4	j = 5
i = 1	-	1	1		
i = 2		-	2	3	
i = 3			-	3	3
i = 4				-	4
i = 5					-

p0	p1	p2	p3	p4	p5
3	4	5	2	7	2

i, j

$$M[1, 3] = \text{Min} \{ M[1, 1] + M[2, 3] + p_0 * p_1 * p_3 \{k=1\}; M[1, 2] + M[3, 3] + p_0 * p_2 * p_3 \{k=2\} \}$$

$$= \text{Min} \{ 0 + 40 + 3 * 4 * 2 \{k=1\}; 60 + 0 + 3 * 5 * 2 \{k=2\} \}$$

$$= \text{Min} \{ 64 \{k=1\}, 90 \{k=2\} \} = 64 \{k=1\}$$

i, j

$$M[2, 4] = \text{Min} \{ M[2, 2] + M[3, 4] + p_1 * p_2 * p_4 \{k=2\}; M[2, 3] + M[4, 4] + p_1 * p_3 * p_4 \{k=3\} \}$$

$$= \text{Min} \{ 0 + 70 + 4 * 5 * 7 \{k=2\}; 40 + 0 + 4 * 2 * 7 \{k=3\} \}$$

$$= \text{Min} \{ 210 \{k=2\}; 96 \{k=3\} \} = 96 \{k=3\}$$

i, j

$$M[3, 5] = \text{Min} \{ M[3, 3] + M[4, 5] + p_2 * p_3 * p_5 \{k=3\}; M[3, 4] + M[5, 5] + p_2 * p_4 * p_5 \{k=4\} \}$$

$$= \text{Min} \{ 0 + 28 + 5 * 2 * 2 \{k=3\}; 70 + 0 + 5 * 7 * 5 \{k=4\} \}$$

$$= \text{Min} \{ 48 \{k=3\}; 245 \{k=4\} \} = 48 \{k=3\}$$

$$M[i, j] = \begin{cases} 0 & \text{if } i = j \\ \text{Min}_{i \leq k < j} & \left(M[i, k] + M[k+1, j] + p_{i-1} * p_k * p_j \right) \end{cases}$$

EXAMPLE 1

A1	p0 x p1	3x4
A2	p1 x p2	4x5
A3	p2 x p3	5x2
A4	p3 x p4	2x7
A5	p4 x p5	7x2

$$M[i, j] = \begin{cases} 0 & \text{if } i = j \\ \text{Min}_{i \leq k < j} & \left(M[i, k] + M[k+1, j] + p_{i-1} * p_k * p_j \right) \end{cases}$$

Spread of '3'

M-table

	j = 1	j = 2	j = 3	j = 4	j = 5
i = 1	0	60	64	106	
i = 2		0	40	96	84
i = 3			0	70	48
i = 4				0	28
i = 5					0

k-table

	j = 1	j = 2	j = 3	j = 4	j = 5
i = 1	-	1	1	3	
i = 2		-	2	3	3
i = 3			-	3	3
i = 4				-	4
i = 5					-

p0	p1	p2	p3	p4	p5
3	4	5	2	7	2

i, j
 $M[1, 4]$
 $= \text{Min} \{ M[1, 1] + M[2, 4] + p_0 * p_1 * p_4 \{k=1\}; M[1, 2] + M[3, 4] + p_0 * p_2 * p_4 \{k=2\};$
 $M[1, 3] + M[4, 4] + p_0 * p_3 * p_4 \{k=3\} \}$
 $= \text{Min} \{ 0 + 96 + 3 * 4 * 7 \{k=1\}; 60 + 70 + 3 * 5 * 7 \{k=2\}; 64 + 0 + 3 * 2 * 7 \{k=3\} \}$
 $= \text{Min} \{ 180 \{k=1\}; 235 \{k=2\}; 106 \{k=3\} \} = 106 \{k=3\}$

i, j
 $M[2, 5]$
 $= \text{Min} \{ M[2, 2] + M[3, 5] + p_1 * p_2 * p_5 \{k=2\}; M[2, 3] + M[4, 5] + p_1 * p_3 * p_5 \{k=3\};$
 $M[2, 4] + M[5, 5] + p_1 * p_4 * p_5 \{k=4\} \}$
 $= \text{Min} \{ 0 + 48 + 4 * 5 * 2 \{k=2\}; 40 + 28 + 4 * 2 * 2 \{k=3\}; 96 + 0 + 4 * 7 * 2 \{k=5\} \}$
 $= \text{Min} \{ 88 \{k=2\}; 84 \{k=3\}; 152 \{k=5\} \} = 84 \{k=3\}$

$$M[i, j] = \begin{cases} 0 & \text{if } i = j \\ \text{Min}_{i \leq k < j} & \left(M[i, k] + M[k+1, j] + p_{i-1} * p_k * p_j \right) \end{cases}$$

EXAMPLE 1

A1	p0 x p1	3x4
A2	p1 x p2	4x5
A3	p2 x p3	5x2
A4	p3 x p4	2x7
A5	p4 x p5	7x2

$$M[i, j] = \begin{cases} 0 & \text{if } i = j \\ \text{Min}_{i \leq k < j} & \left(M[i, k] + M[k+1, j] + p_{i-1} * p_k * p_j \right) \end{cases}$$

Spread of '4'

M-table

	j = 1	j = 2	j = 3	j = 4	j = 5
i = 1	0	60	64	106	104
i = 2		0	40	96	84
i = 3			0	70	48
i = 4				0	28
i = 5					0

k-table

	j = 1	j = 2	j = 3	j = 4	j = 5
i = 1	-	1	1	3	3
i = 2		-	2	3	3
i = 3			-	3	3
i = 4				-	4
i = 5					-

p0	p1	p2	p3	p4	p5
3	4	5	2	7	2

$$M[i, j] = \text{Min}_{i, j} \left\{ \begin{aligned} &M[1, 1] + M[2, 5] + p_0 * p_1 * p_5 \quad \{k=1\}; \\ &M[1, 2] + M[3, 5] + p_0 * p_2 * p_5 \quad \{k=2\}; \\ &M[1, 3] + M[4, 5] + p_0 * p_3 * p_5 \quad \{k=3\}; \\ &M[1, 4] + M[5, 5] + p_0 * p_4 * p_5 \quad \{k=4\} \end{aligned} \right\}$$

$$M[i, j] = \begin{cases} 0 & \text{if } i = j \\ \text{Min}_{i \leq k < j} & \left(M[i, k] + M[k+1, j] + p_{i-1} * p_k * p_j \right) \end{cases}$$

$$= \text{Min} \left\{ \begin{aligned} &0 + 84 + 3 * 4 * 2 \quad \{k=1\}; \\ &60 + 48 + 3 * 5 * 2 \quad \{k=2\}; \\ &64 + 28 + 3 * 2 * 2 \quad \{k=3\}; \\ &106 + 0 + 3 * 7 * 2 \quad \{k=4\} \end{aligned} \right\}$$

$$= \text{Min} \left\{ 108 \{k=1\}; \quad 138 \{k=2\}; \quad 104 \{k=3\}; \quad 148 \{k=4\} \right\}$$

$$= 104 \{k=3\}$$

Final Parenthesisation

$$A1 \times \dots \times A5 = (A1 \times A2 \times A3) * (A4 \times A5)$$

$$(A1 * (A2 \times A3)) * (A4 \times A5)$$

Cross checking:

A2 x A3 requires 4*5*2 = 40 multiplications and results in a 4x2 matrix

A1 * (A2 x A3) requires 3*4*2 = 24 multiplications and results in a 3x2 matrix

A4 x A5 requires 2*7*2 = 28 multiplications and results in a 2x2 matrix

(A1 * (A2 x A3)) * (A4 x A5) requires 3*2*2 = 12 multiplications; results in a 3x2 final product matrix

Total number of multiplications = 40 + 24 + 28 + 12 = 104

EXAMPLE 2

Consider a sequence of five matrices A1, A2, ..., A6 with the dimension parameter 'p' values as follows:

$$M[i, j] = \begin{cases} 0 & \text{if } i = j \\ \text{Min}_{i \leq k < j} \left(M[i, k] + M[k+1, j] + p_{i-1} * p_k * p_j \right) & \text{if } i < j \end{cases}$$

p0	p1	p2	p3	p4	p5	p6
4	10	3	12	20	7	14

A1	4x10
A2	10x3
A3	3x12
A4	12x20
A5	20x7
A6	7x14

Initialization

M-table

	j = 1	j = 2	j = 3	j = 4	j = 5	j = 6
i = 1	0					
i = 2		0				
i = 3			0			
i = 4				0		
i = 5					0	
i = 6						0

k-table

	j = 1	j = 2	j = 3	j = 4	j = 5	j = 6
i = 1	0					
i = 2		0				
i = 3			0			
i = 4				0		
i = 5					0	
i = 6						0

Spread of '1'

- A1 4x10
- A2 10x3
- A3 3x12
- A4 12x20
- A5 20x7
- A6 7x14

M-table

	j = 1	j = 2	j = 3	j = 4	j = 5	j = 6
i = 1	0	120				
i = 2		0	360			
i = 3			0	720		
i = 4				0	1680	
i = 5					0	1960
i = 6						0

k-table

	j = 1	j = 2	j = 3	j = 4	j = 5	j = 6
i = 1	-	1				
i = 2		-	2			
i = 3			-	3		
i = 4				-	4	
i = 5					-	5
i = 6						-

- p0 4
- p1 10
- p2 3
- p3 12
- p4 20
- p5 7
- p6 14

	i, j			
$M[1, 2]$	$= M[1, 1] + M[2, 2] + p_0 \cdot p_1 \cdot p_2$	{k=1}	$= 0 + 0 + 4 \cdot 10 \cdot 3 = 120$	{k=1}
$M[2, 3]$	$= M[2, 2] + M[3, 3] + p_1 \cdot p_2 \cdot p_3$	{k=2}	$= 0 + 0 + 10 \cdot 3 \cdot 12 = 360$	{k=2}
$M[3, 4]$	$= M[3, 3] + M[4, 4] + p_2 \cdot p_3 \cdot p_4$	{k=3}	$= 0 + 0 + 3 \cdot 12 \cdot 20 = 720$	{k=3}
$M[4, 5]$	$= M[4, 4] + M[5, 5] + p_3 \cdot p_4 \cdot p_5$	{k=4}	$= 0 + 0 + 12 \cdot 20 \cdot 7 = 1680$	{k=4}
$M[5, 6]$	$= M[5, 5] + M[6, 6] + p_4 \cdot p_5 \cdot p_6$	{k=5}	$= 0 + 0 + 20 \cdot 7 \cdot 14 = 1960$	{k=5}

$$M[i, j] = \begin{cases} 0 & \text{if } i = j \\ \text{Min}_{i \leq k < j} & \left[M[i, k] + M[k+1, j] + p_{i-1} \cdot p_k \cdot p_j \right] \end{cases}$$

Spread of '2'

- A1 4x10
- A2 10x3
- A3 3x12
- A4 12x20
- A5 20x7
- A6 7x14

M-table

	j = 1	j = 2	j = 3	j = 4	j = 5	j = 6
i = 1	0	120	264			
i = 2		0	360	1320		
i = 3			0	720	1140	
i = 4				0	1680	2856
i = 5					0	1960
i = 6						0

k-table

	j = 1	j = 2	j = 3	j = 4	j = 5	j = 6
i = 1	-	1	2			
i = 2		-	2	2		
i = 3			-	3	4	
i = 4				-	4	5
i = 5					-	5
i = 6						-

- p0 4
- p1 10
- p2 3
- p3 12
- p4 20
- p5 7
- p6 14

$$\begin{aligned}
 M[1, 3] &= \text{Min} \{ M[1, 1] + M[2, 3] + p_0 * p_1 * p_3 \{k=1\}; \\
 &= \text{Min} \{ 0 + 360 + 4 * 10 * 12 \{k=1\}; \\
 &= \text{Min} \{ 840 \{k=1\}; \quad 264 \{k=2\} \} = 264 \{k=2\}
 \end{aligned}$$

$$\begin{aligned}
 &M[1, 2] + M[3, 3] + p_0 * p_2 * p_3 \{k=2\} \} \\
 &120 + 0 + 4 * 3 * 12 \{k=2\} \}
 \end{aligned}$$

$$\begin{aligned}
 M[2, 4] &= \text{Min} \{ M[2, 2] + M[3, 4] + p_1 * p_2 * p_4 \{k=2\}; \\
 &= \text{Min} \{ 0 + 720 + 10 * 3 * 20 \{k=2\}; \\
 &= \text{Min} \{ 1320 \{k=2\}; \quad 2760 \{k=3\} \} = 1320 \{k=2\}
 \end{aligned}$$

$$\begin{aligned}
 &M[2, 3] + M[4, 4] + p_1 * p_3 * p_4 \{k=3\} \} \\
 &360 + 0 + 10 * 12 * 20 \{k=3\} \}
 \end{aligned}$$

$$\begin{aligned}
 M[3, 5] &= \text{Min} \{ M[3, 3] + M[4, 5] + p_2 * p_3 * p_5 \{k=3\}; \\
 &= \text{Min} \{ 0 + 1680 + 3 * 12 * 7 \{k=3\}; \\
 &= \text{Min} \{ 1932 \{k=3\}; \quad 1140 \{k=4\} \} = 1140 \{k=4\}
 \end{aligned}$$

$$\begin{aligned}
 &M[3, 4] + M[5, 5] + p_2 * p_4 * p_5 \{k=4\} \} \\
 &720 + 0 + 3 * 20 * 7 \{k=4\} \}
 \end{aligned}$$

$$\begin{aligned}
 M[4, 6] &= \text{Min} \{ M[4, 4] + M[5, 6] + p_3 * p_4 * p_6 \{k=4\}; \\
 &= \text{Min} \{ 0 + 1960 + 12 * 20 * 14 \{k=4\}; \\
 &= \text{Min} \{ 5320 \{k=4\}; \quad 2856 \{k=5\} \} = 2856 \{k=5\}
 \end{aligned}$$

$$\begin{aligned}
 &M[4, 5] + M[6, 6] + p_3 * p_5 * p_6 \{k=5\} \} \\
 &1680 + 0 + 12 * 7 * 14 \{k=5\} \}
 \end{aligned}$$

$$M[i, j] = \begin{cases} 0 & \text{if } i = j \\ \text{Min}_{i \leq k < j} \left(M[i, k] + M[k+1, j] + p_{i-1} * p_k * p_j \right) & \end{cases}$$

Spread of '3'

- A1 4x10
- A2 10x3
- A3 3x12
- A4 12x20
- A5 20x7
- A6 7x14

M-table

	j = 1	j = 2	j = 3	j = 4	j = 5	j = 6
i = 1	0	120	264	1080		
i = 2		0	360	1320	1350	
i = 3			0	720	1140	1980
i = 4				0	1680	2856
i = 5					0	1960
i = 6						0

k-table

	j = 1	j = 2	j = 3	j = 4	j = 5	j = 6
i = 1	-	1	2	2		
i = 2		-	2	2	2	
i = 3			-	3	4	5
i = 4				-	4	5
i = 5					-	5
i = 6						-

- p0 4
- p1 10
- p2 3
- p3 12
- p4 20
- p5 7
- p6 14

$$\begin{aligned}
 M[1, 4] &= \text{Min}_{i,j} \{ M[1, 1] + M[2, 4] + p_0 \cdot p_1 \cdot p_4 \quad \{k=1\}; \quad M[1, 2] + M[3, 4] + p_0 \cdot p_2 \cdot p_4 \quad \{k=2\}; \\
 &\quad M[1, 3] + M[4, 4] + p_0 \cdot p_3 \cdot p_4 \quad \{k=3\} \} \\
 &= \text{Min} \{ 0 + 1320 + 4 \cdot 10 \cdot 20 \quad \{k=1\}; \quad 120 + 720 + 4 \cdot 3 \cdot 20 \quad \{k=2\}; \quad 264 + 0 + 4 \cdot 12 \cdot 20 \quad \{k=3\} \} \\
 &= \text{Min} \{ 2120 \quad \{k=1\}; \quad 1080 \quad \{k=2\}; \quad 1224 \quad \{k=3\} \} = 1080 \quad \{k=2\}
 \end{aligned}$$

$$\begin{aligned}
 M[2, 5] &= \text{Min}_{i,j} \{ M[2, 2] + M[3, 5] + p_1 \cdot p_2 \cdot p_5 \quad \{k=2\}; \quad M[2, 3] + M[4, 5] + p_1 \cdot p_3 \cdot p_5 \quad \{k=3\}; \\
 &\quad M[2, 4] + M[5, 5] + p_1 \cdot p_4 \cdot p_5 \quad \{k=4\} \} \\
 &= \text{Min} \{ 0 + 1140 + 10 \cdot 3 \cdot 7 \quad \{k=2\}; \quad 360 + 1680 + 10 \cdot 12 \cdot 7 \quad \{k=3\}; \quad 1320 + 0 + 10 \cdot 20 \cdot 7 \quad \{k=4\} \} \\
 &= \text{Min} \{ 1350 \quad \{k=2\}; \quad 2880 \quad \{k=3\}; \quad 2720 \quad \{k=4\} \} = 1350 \quad \{k=2\}
 \end{aligned}$$

$$\begin{aligned}
 M[3, 6] &= \text{Min}_{i,j} \{ M[3, 3] + M[4, 6] + p_2 \cdot p_3 \cdot p_6 \quad \{k=3\}; \quad M[3, 4] + M[5, 6] + p_2 \cdot p_4 \cdot p_6 \quad \{k=4\}; \\
 &\quad M[3, 5] + M[6, 6] + p_2 \cdot p_5 \cdot p_6 \quad \{k=5\} \} \\
 &= \text{Min} \{ 0 + 2856 + 3 \cdot 7 \cdot 14 \quad \{k=3\}; \quad 720 + 1960 + 3 \cdot 20 \cdot 14 \quad \{k=4\}; \quad 1140 + 0 + 3 \cdot 20 \cdot 14 \quad \{k=5\} \} \\
 &= \text{Min} \{ 3150 \quad \{k=3\}; \quad 3520 \quad \{k=4\}; \quad 1980 \quad \{k=5\} \} = 1980 \quad \{k=5\}
 \end{aligned}$$

$$M[i, j] = \begin{cases} 0 & \text{if } i = j \\ \text{Min}_{i \leq k < j} \left(M[i, k] + M[k+1, j] + p_{i-1} \cdot p_k \cdot p_j \right) & \end{cases}$$

Spread of '4'

- A1 4x10
- A2 10x3
- A3 3x12
- A4 12x20
- A5 20x7
- A6 7x14

M-table

	j = 1	j = 2	j = 3	j = 4	j = 5	j = 6
i = 1	0	120	264	1080	1344	
i = 2		0	360	1320	1350	2330
i = 3			0	720	1140	1980
i = 4				0	1680	2856
i = 5					0	1960
i = 6						0

k-table

	j = 1	j = 2	j = 3	j = 4	j = 5	j = 6
i = 1	-	1	2	2	2	
i = 2		-	2	2	2	5
i = 3			-	3	4	5
i = 4				-	4	5
i = 5					-	5
i = 6						-

p0	p1	p2	p3	p4	p5	p6
4	10	3	12	20	7	14

$$\begin{aligned}
 M[1, 5] &= \text{Min} \{ M[1, 1] + M[2, 5] + p_0 \cdot p_1 \cdot p_5 \quad \{k=1\}; \\
 &\quad M[1, 3] + M[4, 5] + p_0 \cdot p_3 \cdot p_5 \quad \{k=3\}; \\
 &= \text{Min} \{ 0 + 1350 + 4 \cdot 10 \cdot 7 \quad \{k=1\}; \\
 &\quad 264 + 1680 + 4 \cdot 12 \cdot 7 \quad \{k=3\}; \\
 &= \text{Min} \{ 1630 \quad \{k=1\}; \quad 1344 \quad \{k=2\}; \quad 2280 \quad \{k=3\}; \\
 &= 1344 \quad \{k=2\}
 \end{aligned}$$

$$\begin{aligned}
 &M[1, 2] + M[3, 5] + p_0 \cdot p_2 \cdot p_5 \quad \{k=2\}; \\
 &M[1, 4] + M[5, 5] + p_0 \cdot p_4 \cdot p_5 \quad \{k=4\} \} \\
 &120 + 1140 + 4 \cdot 3 \cdot 7 \quad \{k=2\}; \\
 &1080 + 0 + 4 \cdot 20 \cdot 7 \quad \{k=4\} \} \\
 &1640 \quad \{k=4\} \}
 \end{aligned}$$

$$\begin{aligned}
 M[2, 6] &= \text{Min} \{ M[2, 2] + M[3, 6] + p_1 \cdot p_2 \cdot p_6 \quad \{k=2\}; \\
 &\quad M[2, 4] + M[5, 6] + p_1 \cdot p_4 \cdot p_6 \quad \{k=4\}; \\
 &= \text{Min} \{ 0 + 1980 + 10 \cdot 3 \cdot 14 \quad \{k=2\}; \\
 &\quad 1320 + 1960 + 10 \cdot 20 \cdot 14 \quad \{k=4\}; \\
 &= \text{Min} \{ 2400 \quad \{k=2\}; \quad 4896 \quad \{k=3\}; \quad 6080 \quad \{k=4\}; \\
 &= 2330 \quad \{k=5\}
 \end{aligned}$$

$$\begin{aligned}
 &M[2, 3] + M[4, 6] + p_1 \cdot p_3 \cdot p_6 \quad \{k=3\}; \\
 &M[2, 5] + M[6, 6] + p_1 \cdot p_5 \cdot p_6 \quad \{k=5\} \} \\
 &360 + 2856 + 10 \cdot 12 \cdot 14 \quad \{k=3\}; \\
 &1350 + 0 + 10 \cdot 7 \cdot 14 \quad \{k=5\} \} \\
 &2330 \quad \{k=5\} \}
 \end{aligned}$$

$$M[i, j] = \begin{cases} 0 & \text{if } i = j \\ \text{Min}_{i \leq k < j} \left(M[i, k] + M[k+1, j] + p_{i-1} \cdot p_k \cdot p_j \right) & \text{otherwise} \end{cases}$$

Spread of '5'

- A1 4x10
- A2 10x3
- A3 3x12
- A4 12x20
- A5 20x7
- A6 7x14

M-table

	j = 1	j = 2	j = 3	j = 4	j = 5	j = 6
i = 1	0	120	264	1080	1344	1736
i = 2		0	360	1320	1350	2330
i = 3			0	720	1140	1980
i = 4				0	1680	2856
i = 5					0	1960
i = 6						0

k-table

	j = 1	j = 2	j = 3	j = 4	j = 5	j = 6
i = 1	-	1	2	2	2	5
i = 2		-	2	2	2	5
i = 3			-	3	4	5
i = 4				-	4	5
i = 5					-	5
i = 6						-

- p0 4
- p1 10
- p2 3
- p3 12
- p4 20
- p5 7
- p6 14

$$M[i, j] = \begin{cases} 0 & \text{if } i = j \\ \text{Min}_{i \leq k < j} \left(M[i, k] + M[k+1, j] + p_{i-1} * p_k * p_j \right) & \text{otherwise} \end{cases}$$

$$M[1, 6] = \text{Min}_{i, j} \left\{ \begin{array}{l} M[1, 1] + M[2, 6] + p_0 * p_1 * p_6 \quad \{k=1\}; \\ M[1, 2] + M[3, 6] + p_0 * p_2 * p_6 \quad \{k=2\}; \\ M[1, 3] + M[4, 6] + p_0 * p_3 * p_6 \quad \{k=3\}; \\ M[1, 4] + M[5, 6] + p_0 * p_4 * p_6 \quad \{k=4\}; \\ M[1, 5] + M[6, 6] + p_0 * p_5 * p_6 \quad \{k=5\} \end{array} \right\}$$

$$= \text{Min} \left\{ \begin{array}{l} 0 + 2330 + 4 * 10 * 14 \quad \{k=1\}; \\ 120 + 1980 + 4 * 3 * 14 \quad \{k=2\}; \\ 264 + 2856 + 4 * 12 * 14 \quad \{k=3\}; \\ 1080 + 1960 + 4 * 20 * 14 \quad \{k=4\}; \\ 1344 + 0 + 4 * 7 * 14 \quad \{k=5\} \end{array} \right\}$$

$$= \text{Min} \left\{ 2890 \{k=1\}; \quad 2268 \{k=2\}; \quad 3792 \{k=3\}; \quad 4160 \{k=4\}; \quad 1736 \{k=5\} \right\}$$

$$= 1736 \{k=5\}$$

$$A1 \times A2 \times A3 \times A4 \times A5 \times A6 = (A1 \times A2 \times A3 \times A4 \times A5) \times A6$$

$$= ((A1 \times A2) * (A3 \times A4 \times A5)) * A6 = ((A1 \times A2) * ((A3 \times A4) * A5)) \times A6$$

Cross-checking:

- A1 x A2 will require 4*10*3 = 120 multiplications and be a 4x3 matrix
- A3 x A4 will require 3*12*20 = 720 multiplications and be a 3x20 matrix
- (A3 x A4) x A5 will require 3*20*7 = 420 multiplications and be a 3x7 matrix
- (A1 x A2) * ((A3 x A4) * A5) will require 4*3*7 = 84 multiplications and be a 4x7 matrix
- ((A1 x A2) * ((A3 x A4) * A5)) x A6 will require 4*7*14 = 392 multiplications and be a 4x14 matrix

$$\text{Total number of multiplications} = 120 + 720 + 420 + 84 + 392 = 1736$$

Examples: Solutions to sub problems

- A1 4x10
- A2 10x3
- A3 3x12
- A4 12x20
- A5 20x7
- A6 7x14

M-table

	j = 1	j = 2	j = 3	j = 4	j = 5	j = 6
i = 1	0	120	264	1080	1344	1736
i = 2		0	360	1320	1350	2330
i = 3			0	720	1140	1980
i = 4				0	1680	2856
i = 5					0	1960
i = 6						0

k-table

	j = 1	j = 2	j = 3	j = 4	j = 5	j = 6
i = 1	-	1	2	2	2	5
i = 2		-	2	2	2	5
i = 3			-	3	4	5
i = 4				-	4	5
i = 5					-	5
i = 6						-

A2 x A3 x A4 x A5: 1350 multiplications

$$A2 \times A3 \times A4 \times A5 = (A2) \times (A3 \times A4 \times A5) = (A2) \times ((A3 \times A4) \times A5)$$

Cross-checking:

A3 x A4 will require $3 \times 12 \times 20 = 720$ multiplications and be a 3x20 matrix

(A3 x A4) x A5 will require $3 \times 20 \times 7 = 420$ multiplications and be a 3x7 matrix

(A2) x ((A3 x A4) x A5) will require $10 \times 3 \times 7 = 210$ multiplications and be a 10x7 matrix

$$\text{Total number of multiplications} = 720 + 420 + 210 = 1350$$