

# Αλγόριθμοι και Πολυπλοκότητα

7ο εξάμηνο  
Σ.Η.Μ.Μ.Υ. & Σ.Ε.Μ.Φ.Ε.

<http://www.corelab.ece.ntua.gr/courses/>

5η εβδομάδα: Δυναμικός Προγραμματισμός (Dynamic Programming): Discrete Knapsack, All-Pairs Shortest Paths, Traveling Salesman

Διδάσκοντες:  
Στάθης Ζάχος - Άρης Παγουρτζής

# Δυναμικός Προγραμματισμός

## Αρχή της Βελτιστότητας (Optimality Principle)

*Οποιαδήποτε υπακολουθία της βέλτιστης ακολουθίας αποφάσεων είναι βέλτιστη για το αντίστοιχο υποπρόβλημα.*

# Παράδειγμα

Συντομότερα μονοπάτια:

Αν  $v \rightarrow w$ :  $v, \dots, u, \dots, w$  συντομότερο  
τότε και  $u \rightarrow w$ :  $u, \dots, w$  συντομότερο

Αναδρομική σχέση «προς τα εμπρός» (forward):

$$MinPathCost_{u \rightarrow w} = \min_{\forall k \text{ adjacent to } u} (Cost(u, k) + MinPathCost_{k \rightarrow w})$$

Αναδρομική σχέση «προς τα πίσω» (backward):

$$MinPathCost_{u \rightarrow w} = \min_{\forall k \text{ adjacent to } w} (MinPathCost_{u \rightarrow k} + Cost(k, w))$$

# Πρόβλημα Σακιδίου Ακεραίων Τιμών (Discrete Knapsack Problem)

Ορισμός: όπως το Knapsack, αλλά κάθε αντικείμενο  $u_i$  μπαίνει στο σακίδιο είτε ολόκληρο ( $x_i=1$ ) είτε καθόλου ( $x_i=0$ ).

$$\begin{cases} \sum w_i x_i & \leq M \\ \sum p_i x_i & = \text{MAXIMUM} \end{cases}$$

# Discrete Knapsack (συν.)

Ισχύει η αρχή βελτιστότητας:

Κάθε τελικό τμήμα

$$x_i, \dots, x_n$$

της βέλτιστης ακολουθίας

$$x_1, x_2, \dots, x_n$$

είναι βέλτιστη ακολουθία για το αντίστοιχο

υποπρόβλημα με βάρος  $M' = M - \sum_{k=1}^{i-1} w_k x_k$

# Αναδρομικές Σχέσεις

Forward approach:

$$Knap(1..n, M) = \max\{0 + Knap(2..n, M), p_1 + Knap(2..n, M - w_1)\}$$

Backward approach

$$Knap(1..n, M) = \max\{0 + Knap(1..n - 1, M), p_n + Knap(1..n - 1, M - w_n)\}$$

Γενικότερα:

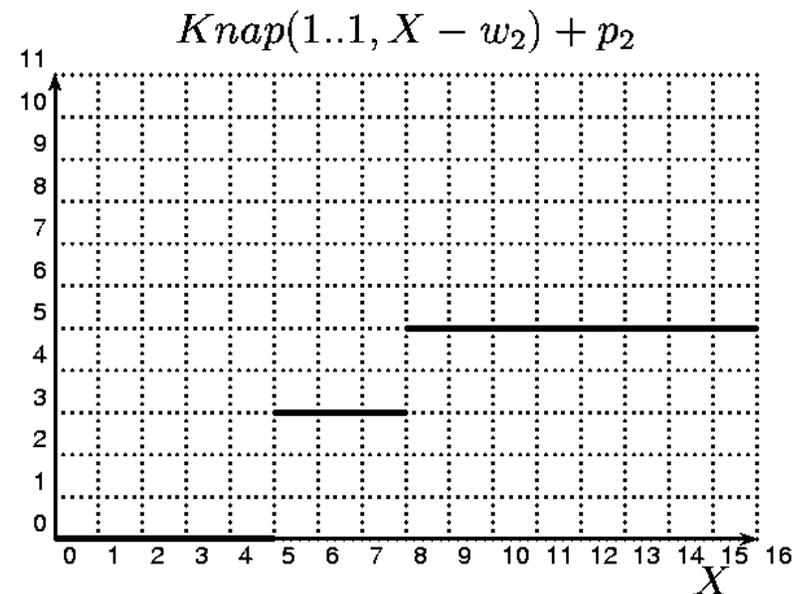
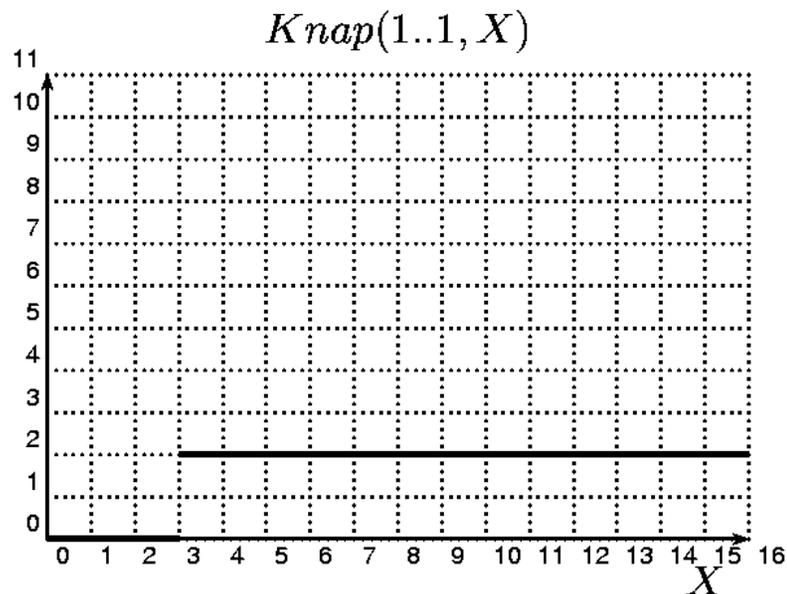
$$Knap(1..i, X) = \max\{0 + Knap(1..i - 1, X), p_i + Knap(1..i - 1, X - w_i)\}$$

# Παράδειγμα

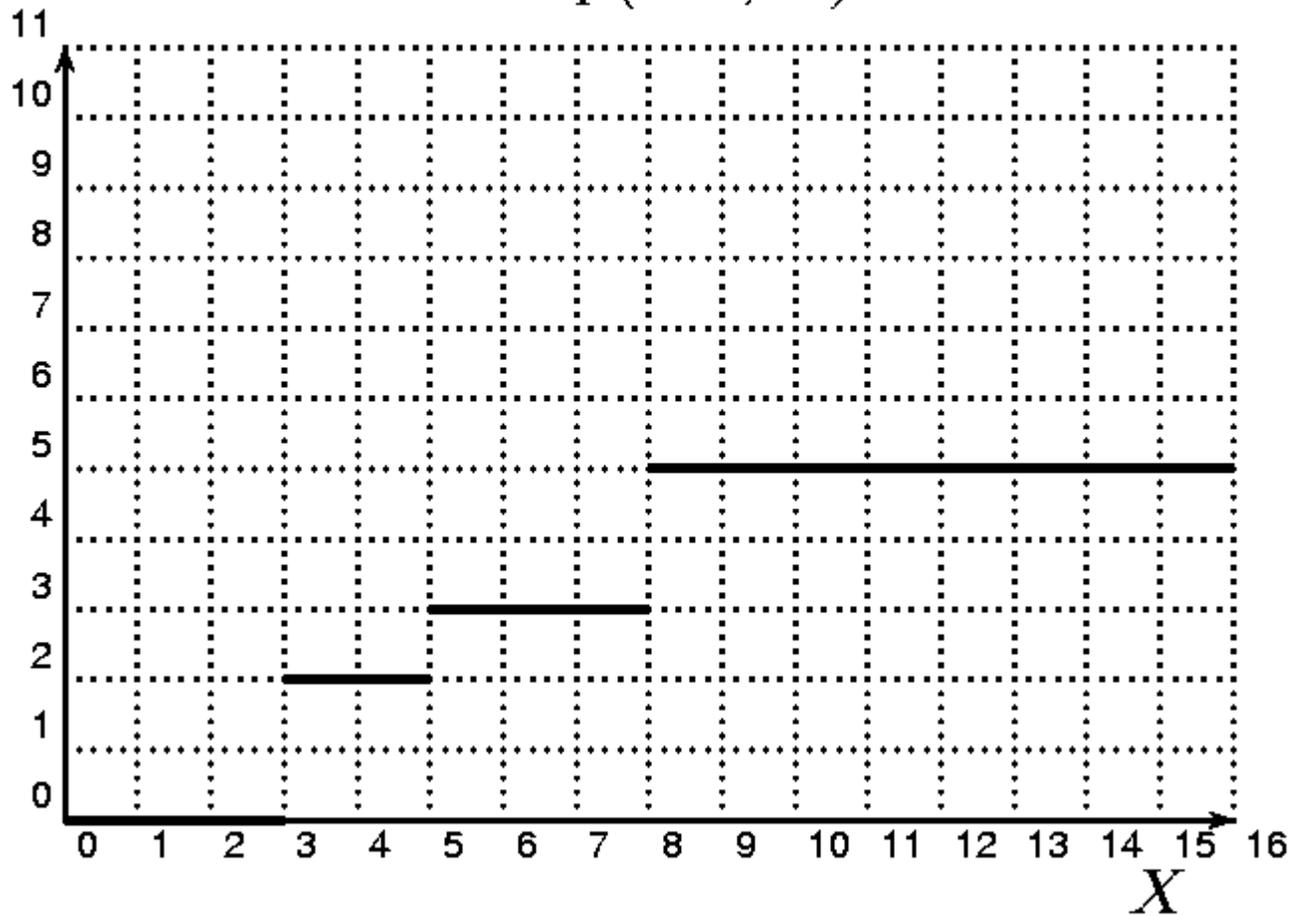
$$w_1 = 3 \quad w_2 = 5 \quad w_3 = 6$$

$$p_1 = 2 \quad p_2 = 3 \quad p_3 = 5$$

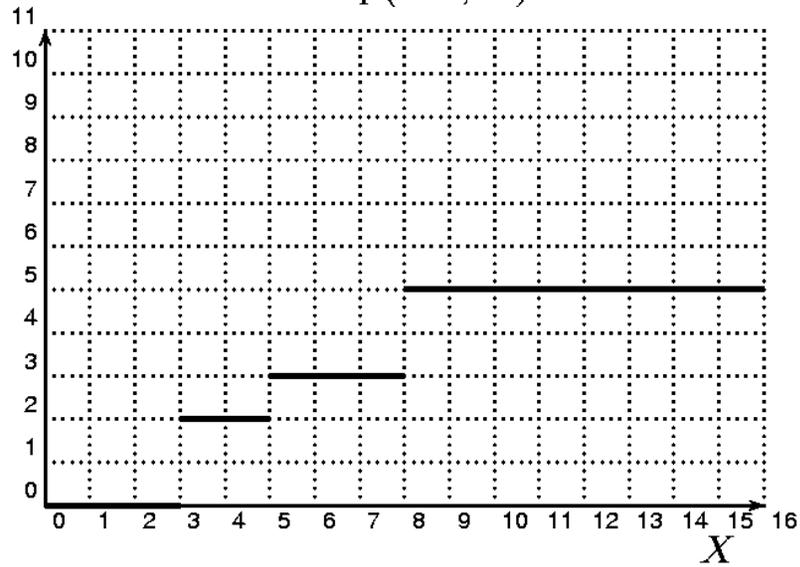
$$M = 10$$



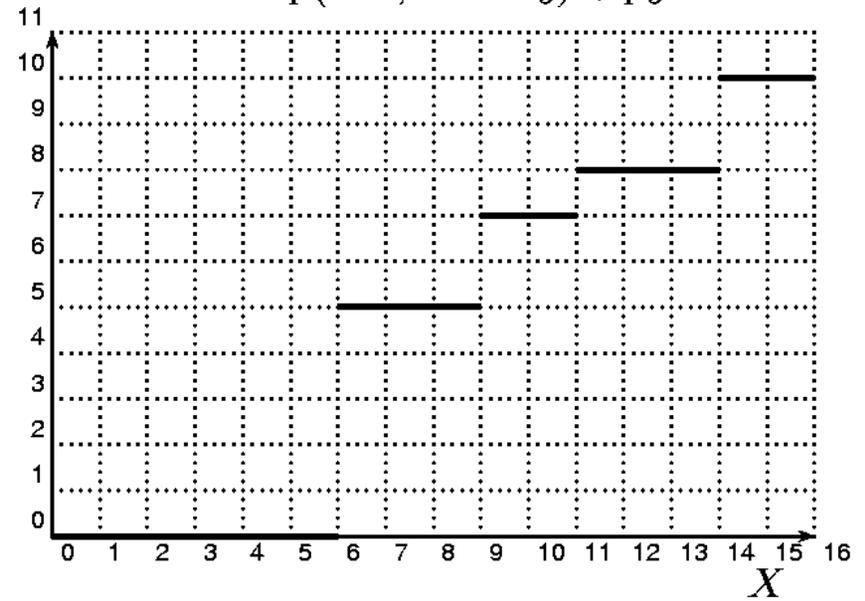
$Knap(1..2, X)$



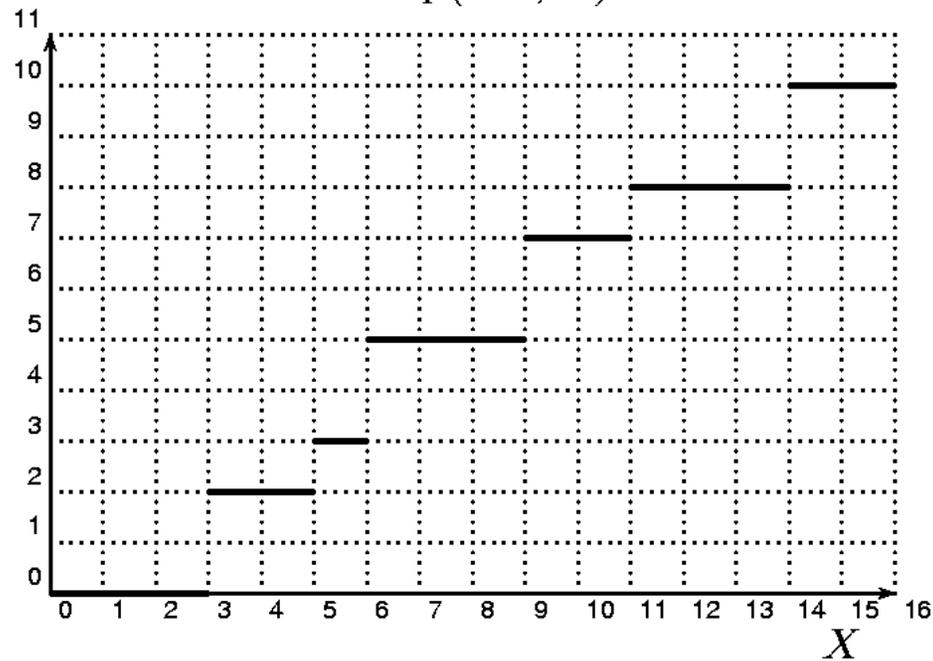
$Knap(1..2, X)$

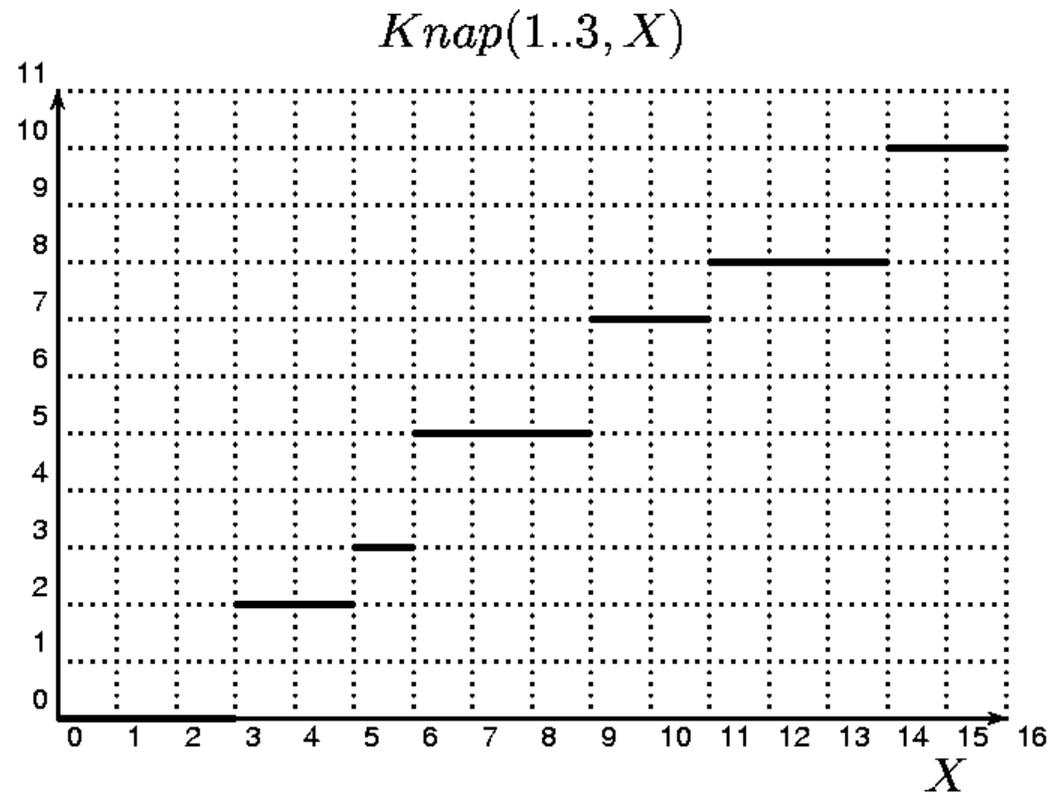


$Knap(1..2, X - w_3) + p_3$



$Knap(1..3, X)$





Σημαντικά ζεύγη:

$(0, 0)$ ,  $(3, 2)$ ,  $(5, 3)$ ,  $(6, 5)$ ,  $(9, 7)$ ,  $(11, 8)$ ,  $(14, 10)$

# Αλγόριθμος Discrete Knapsack

- $S_0^0 = \{(0, 0)\}$
- for  $i:=1$  to  $n$  do
  - $S_1^i = \{(w + w_i, p + p_i) \mid (w, p) \in S_0^{i-1}\}$
  - $S_0^i = \text{merge-discard}(S_0^{i-1}, S_1^i)$
- traceback

-----

$\text{merge-discard}(S_0^{i-1}, S_1^i) =$

$$S_0^{i-1} \cup S_1^i - \{(w, p) \mid \exists (w', p') : w' \leq w \wedge p' \geq p\} - \{(w, p) \mid w > M\}$$

# Παράδειγμα Merge-Discard

$i$	$S_0^i$	$S_1^i$
0	$\{(0,0)\}$	
1		$\{(3,2)\}$
1	$\{(0,0),(3,2)\}$	
2		$\{(5,3),(8,5)\}$
2	$\{(0,0),(3,2),(5,3),(8,5)\}$	
3		$\{(6,5),(9,7),(11,8),(14,10)\}$
3	$\{(0,0),(3,2),(5,3),(8,5),(6,5),(9,7),(11,8),(14,10)\}$	

# Διαδικασία Traceback

```
procedure Traceback;  
begin  
  A:=(the last pair of  $S_0^n$  that is the pair with maximum cost);  
  for  $i := n$  downto 1 do  
    if  $A$  in  $S_1^i$  then begin  $x_i := 1$ ;  $A := (w_A - w_i, p_A - p_i)$  end  
    else  $x_i := 0$   
  end  
end
```

# Πολυπλοκότητα αλγορίθμου Discrete Knapsack

Χειρότερη περίπτωση: εκθετική

Συγκεκριμένα:

$O(\min(2^n, nW))$ , όπου  $W$  το άθροισμα των βαρών.

# All Pairs Shortest Paths

Είσοδος: Γράφος  $G(V,E)$  με  $n$  κόμβους και  
βάρη σε κάθε πλευρά  $u \rightarrow w$ ,  $C[u,w]$ .

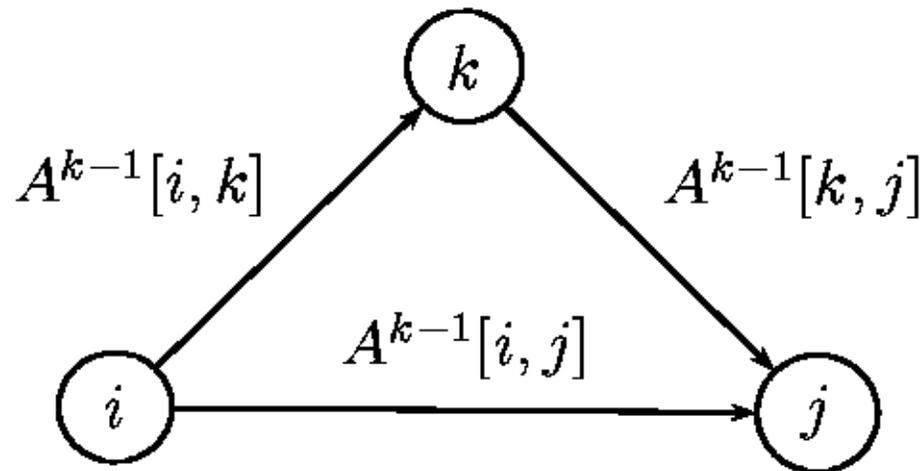
Εξοδος: συντομότερο μονοπάτι για κάθε  
ζεύγος κόμβων.

*Αρχικοποίηση:*

$$A[i, j] = \begin{cases} 0 & , i = j \\ C[i, j] & , i \neq j \& (i, j) \in E \\ \infty & , \text{αλλιώς} \end{cases}$$

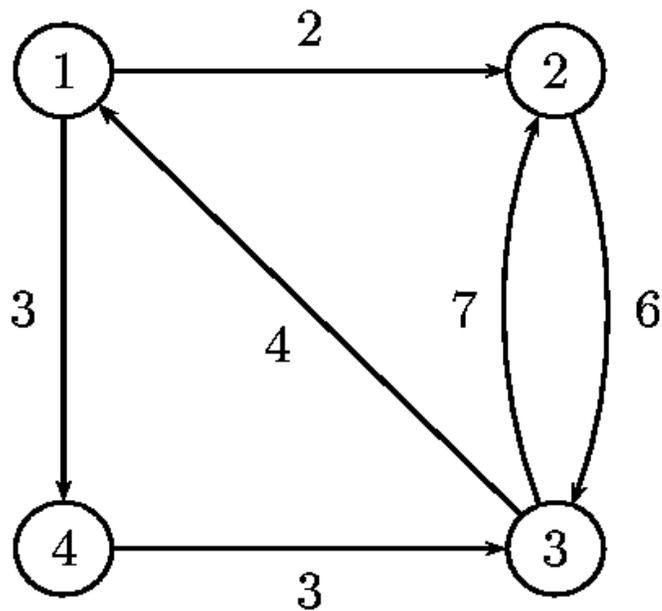
# Αναδρομική σχέση

$$A^k[i, j] = \min(A^{k-1}[i, j], A^{k-1}[i, k] + A^{k-1}[k, j])$$



# Αλγόριθμος Floyd

```
procedure AllShortestPaths (...);  
begin  
  for i:=1 to n do  
    for j:=1 to n do A[i,j]:=Cost[i,j];  
    for k:=1 to n do  
      for i:=1 to n do  
        for j:=1 to n do  
          A[i,j]:= min(A[i,j], A[i,k]+A[k,j])  
    end  
  end  
end
```

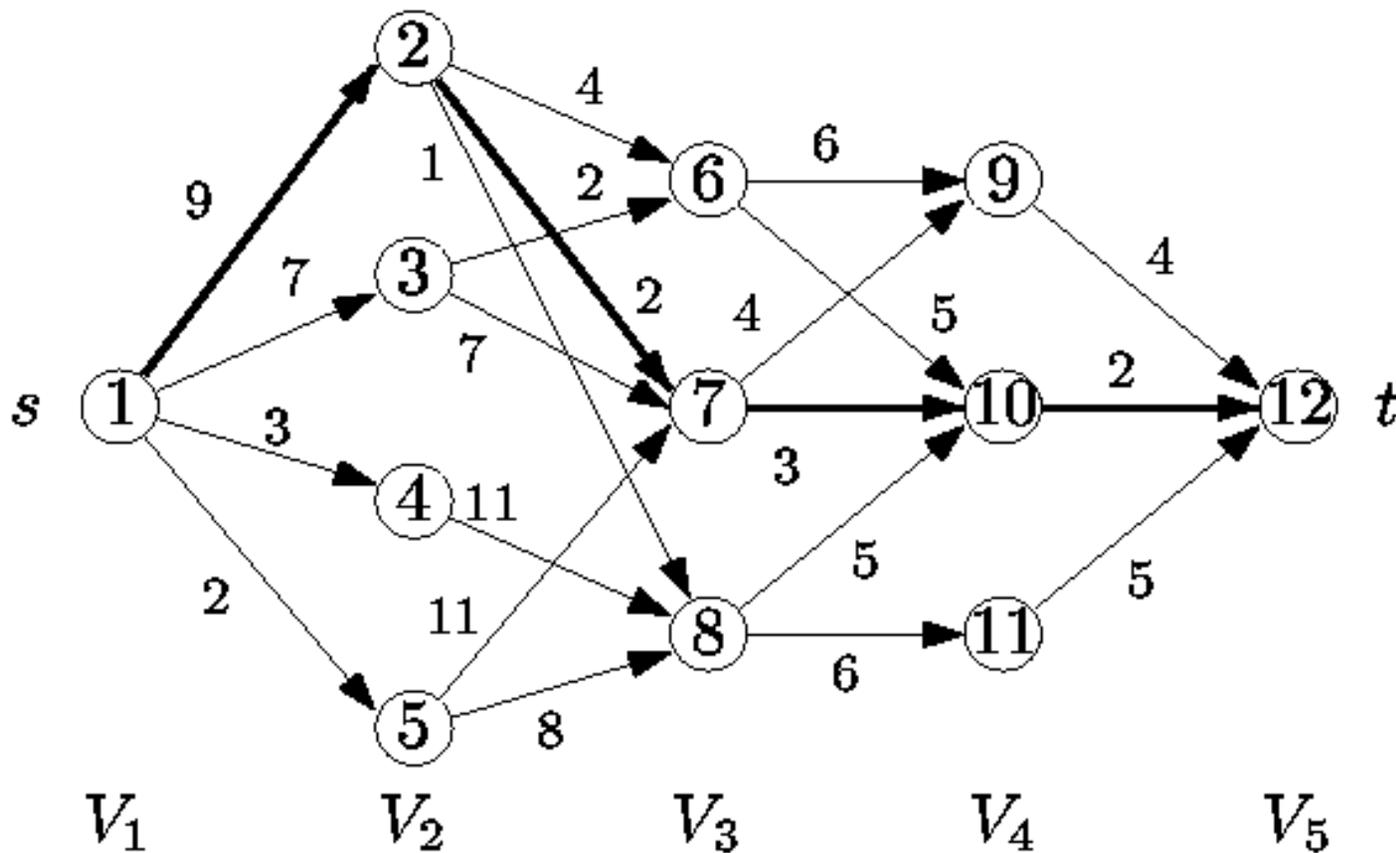


$$C = \begin{bmatrix} 0 & 2 & \infty & 3 \\ \infty & 0 & 6 & \infty \\ 4 & 7 & 0 & \infty \\ \infty & \infty & 3 & 0 \end{bmatrix}$$

$$A^1 = \begin{bmatrix} 0 & 2 & \infty & 3 \\ \infty & 0 & 6 & \infty \\ 4 & 6 & 0 & 7 \\ \infty & \infty & 3 & 0 \end{bmatrix}, A^2 = \begin{bmatrix} 0 & 2 & 8 & 3 \\ \infty & 0 & 6 & \infty \\ 4 & 6 & 0 & 7 \\ \infty & \infty & 3 & 0 \end{bmatrix}$$

$$A^3 = \begin{bmatrix} 0 & 2 & 8 & 3 \\ 10 & 0 & 6 & 13 \\ 4 & 6 & 0 & 7 \\ 7 & 9 & 3 & 0 \end{bmatrix}, A^4 = \begin{bmatrix} 0 & 2 & 6 & 3 \\ 10 & 0 & 6 & 13 \\ 4 & 6 & 0 & 7 \\ 7 & 9 & 3 & 0 \end{bmatrix}$$

# Συντομότερο Μονοπάτι σε multistage graph



# Αλγόριθμος για shortest path σε multistage graph

*Αναδρομική σχέση:*

$$OptCost[1, n] = \min_{r \in V_2} \{cost[1, r] + OptCost[r, n]\}$$

---

```
procedure ShortestPathInMultistageGraph (...)  
begin  
  DistFromN[n]:=0;  
  for j:=n-1 downto 1 do  
    begin  
      select r with min{cost[j,r]+DistFromN[r]};  
      DistFromN[j]:=cost[j,r]+DistFromN[r]; next[j]:=r;  
    end  
  end  
end
```

---

# Πρόβλημα Πλανόδιου Πωλητή (Traveling Salesman Problem - TSP)

*Είσοδος:* Πλήρης γράφος με βάρη.

*Εξοδος:* Κύκλος ελαχίστου κόστους που περνάει από όλους τους κόμβους 1 φορά.

Αρχή βελτιστότητας:

$$MinCostTour_{1 \rightarrow 1} = \min_{k \neq 1} \{cost[1, k] + MinCostTour_{k \rightarrow 1}\}$$

# Αναδρομικές Σχέσεις

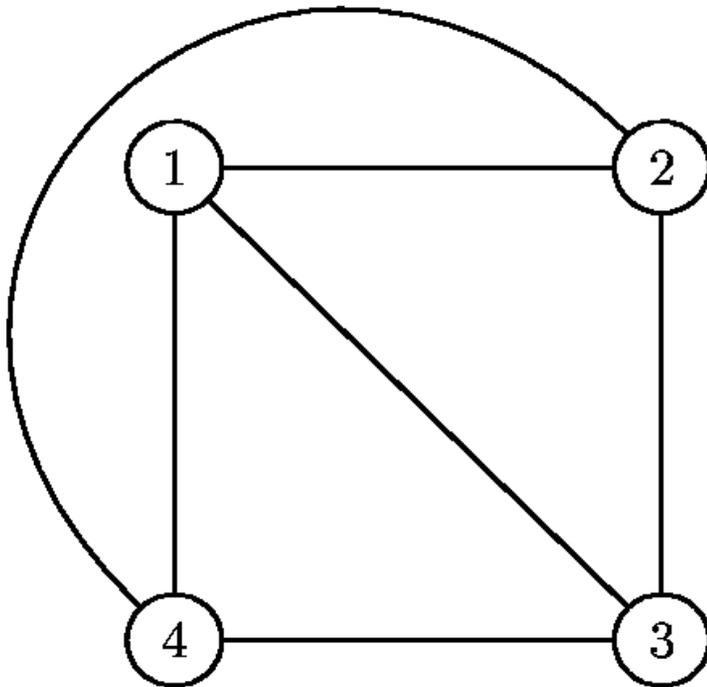
$g(i, S)$  : κόστος συντομότερου μονοπατιού από τον  $i$  στον  $1$  που περνά από όλους τους κόμβους του  $S$ .

$$g(1, V - \{1\}) = \min_{2 \leq k \leq n} \{cost[1, k] + g(k, V - \{1, k\})\}$$

$$g(i, S) = \min_{j \in S} \{cost[i, j] + g(j, S - \{j\})\}$$

$$g(i, \emptyset) = cost[i, 1].$$

# Παράδειγμα



$$C = \begin{bmatrix} 0 & 10 & 15 & 20 \\ 5 & 0 & 9 & 10 \\ 6 & 13 & 0 & 12 \\ 8 & 8 & 9 & 0 \end{bmatrix}$$

# Επίλυση

$|V|=0$

$$g(2, \emptyset) = Cost[2, 1] = 5$$

$$g(3, \emptyset) = Cost[3, 1] = 6$$

$$g(4, \emptyset) = Cost[4, 1] = 8$$

$|V|=1$

$$g(2, \{3\}) = Cost[2, 3] + g(3, \emptyset) = 9 + 6 = 15$$

$$g(2, \{4\}) = Cost[2, 4] + g(4, \emptyset) = 10 + 8 = 18$$

$$g(3, \{2\}) = Cost[3, 2] + g(2, \emptyset) = 13 + 5 = 18$$

$$g(3, \{4\}) = Cost[3, 4] + g(4, \emptyset) = 12 + 8 = 20$$

$$g(4, \{2\}) = Cost[4, 2] + g(2, \emptyset) = 8 + 5 = 13$$

$$g(4, \{3\}) = Cost[4, 3] + g(3, \emptyset) = 9 + 6 = 15$$

$|V|=2$

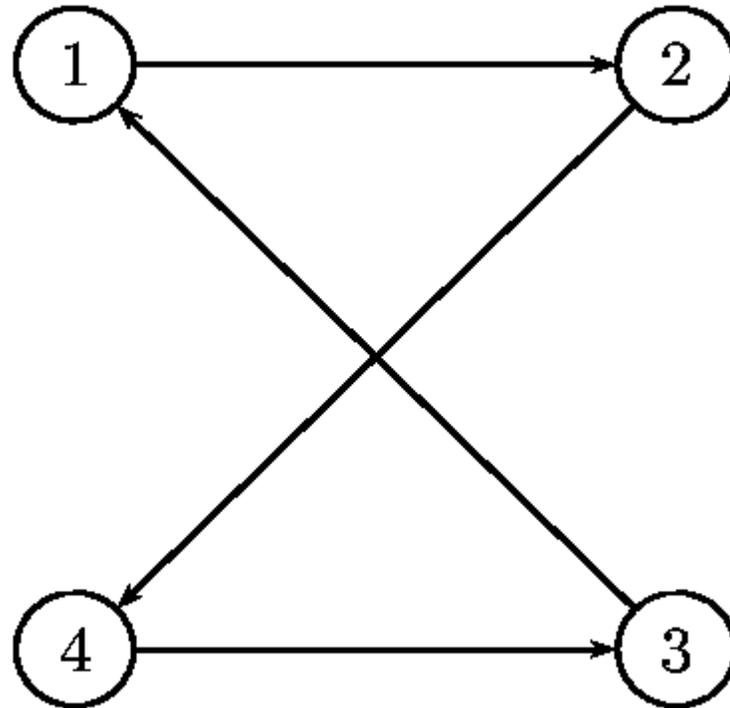
$$\begin{aligned}g(2, \{3, 4\}) &= \min(\text{Cost}[2, 3] + g(3, \{4\}), \text{Cost}[2, 4] + g(4, \{3\})) \\ &= \min(9 + 20, 10 + 15) = 25\end{aligned}$$

$$\begin{aligned}g(3, \{2, 4\}) &= \min(\text{Cost}[3, 2] + g(2, \{4\}), \text{Cost}[3, 4] + g(4, \{2\})) \\ &= \min(13 + 18, 12 + 13) = 25\end{aligned}$$

$$\begin{aligned}g(4, \{2, 3\}) &= \min(\text{Cost}[4, 2] + g(2, \{3\}), \text{Cost}[4, 3] + g(3, \{2\})) \\ &= \min(8 + 15, 9 + 18) = 23\end{aligned}$$

$|V|=3$

$$\begin{aligned}g(1, \{2, 3, 4\}) &= \min(\text{Cost}[1, 2] + g(2, \{3, 4\}), \text{Cost}[1, 3] + g(3, \{2, 4\}), \\ &\quad \text{Cost}[1, 4] + g(4, \{2, 3\})) \\ &= \min(10 + 25, 15 + 25, 20 + 23) = 35\end{aligned}$$



$$Space = (n - 1) \sum_{k=0}^{n-2} \binom{n-2}{k} = (n - 1)(1 + 1)^{n-2} = (n - 1) \cdot 2^{n-2}$$