

# Συντομότερες Διαδρομές

---

Διδάσκοντες:

**Αρ. Παγουρτζής, Δ. Φωτάκης, Δ. Σούλιου**

Επιμέλεια διαφανειών: **Δ. Φωτάκης**

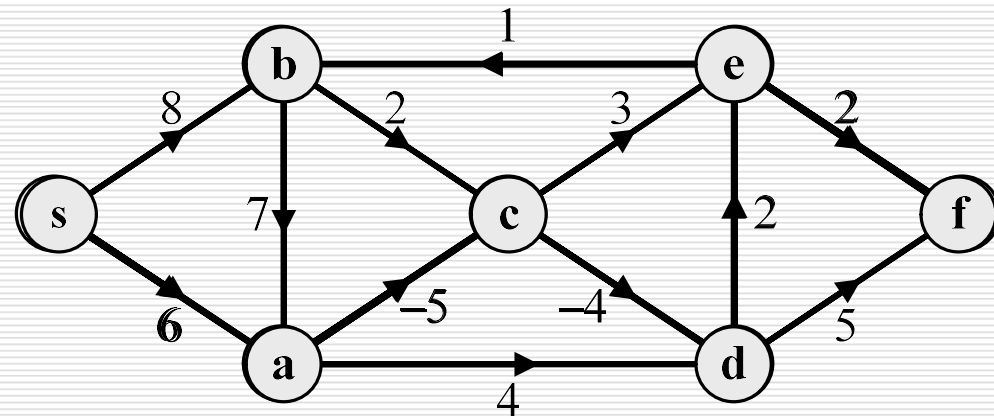
Σχολή Ηλεκτρολόγων Μηχανικών  
και Μηχανικών Υπολογιστών

Εθνικό Μετσόβιο Πολυτεχνείο



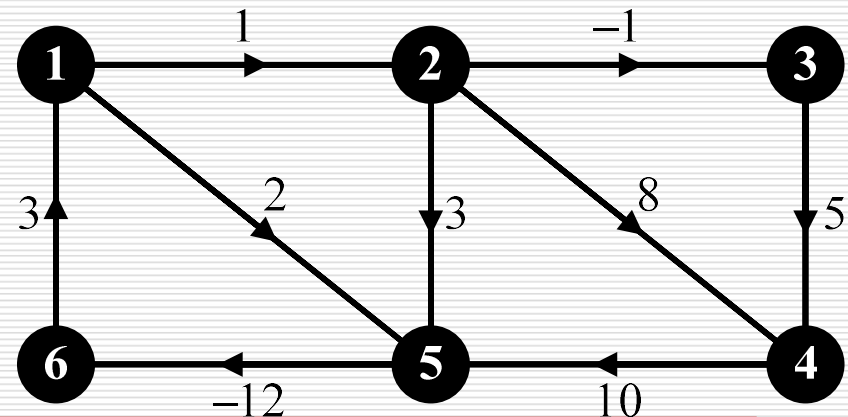
# Συντομότερη Διαδρομή

- Κατευθυνόμενο  $G(V, E, w)$  με μήκη  $w : E \mapsto \mathbb{R}$ 
  - Μήκος διαδρομής  $p = (v_0, v_1, \dots, v_k) : w(p) = \sum_{i=1}^k w(v_{i-1}, v_i)$
  - Απόσταση  $d(u, v)$ : μήκος συντομότερης  $u - v$  διαδρομής.
  - Αν δεν υπάρχει  $u - v$  διαδρομή,  $d(u, v) = \infty$ .
- Ζητούμενο: αποστάσεις και συντομότερες διαδρομές από **αρχική κορυφή**  $s$  προς **όλες** τις κορυφές.
  - Θεμελιώδες πρόβλημα συνδυαστικής βελτιστοποίησης.



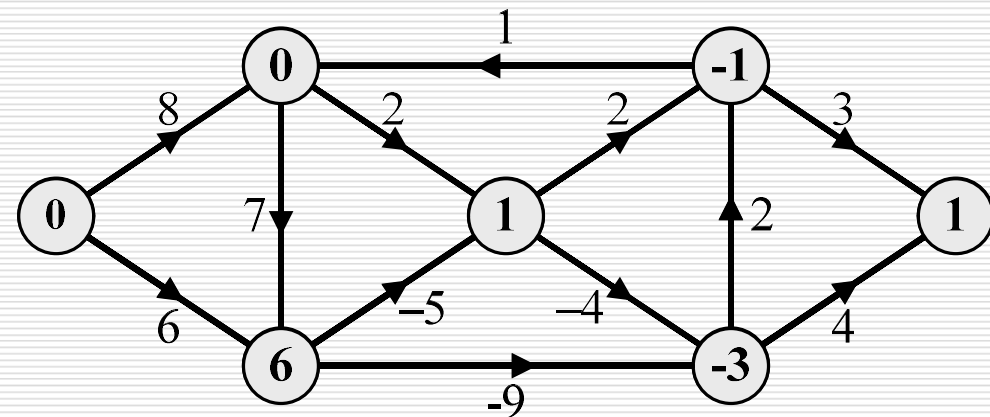
# Κύκλοι Αρνητικού Μήκους

- **Διαδρομή:** ακολ. κορυφών όπου διαδοχικές συνδέονται με ακμή.
- **Μονοκονδυλιά:** διαδρομή χωρίς επαναλαμβανόμενες ακμές.
- **(Απλό) μονοπάτι:** διαδρομή χωρίς επαναλαμβανόμενες κορυφές.
  - Υπάρχει διαδρομή  $u - v$  αν υπάρχει **μονοπάτι**  $u - v$ .
- **Συντομότερη** διαδρομή είναι **μονοπάτι** εκτός αν...
  - Υπάρχει **κύκλος αρνητικού μήκους!**
  - Αποστάσεις **δεν ορίζονται** γιατί συνολικό μήκος διαδρομής μπορεί να **μειώνεται επ' άπειρο!**
  - Κύκλος αρνητικού μήκους σε κάποια  $u - v$  διαδρομή  $\Rightarrow d(u, v) = -\infty$ .



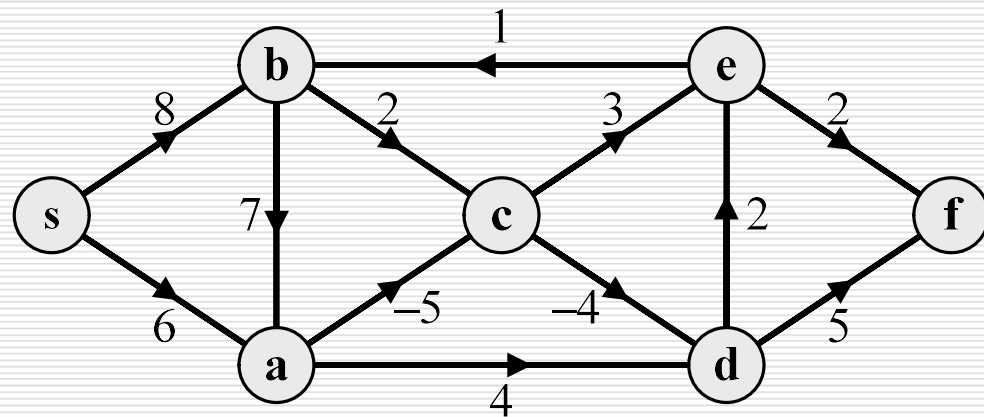
# Συντομότερα Μονοπάτια

- Αν  $p = (v_0, v_1, \dots, v_k)$  είναι **συντομότερο** μονοπάτι, **κάθε**  $v_i - v_j$  τμήμα του αποτελεί **συντομότερο**  $v_i - v_j$  μονοπάτι.
  - **Αρχή βελτιστότητας.**
- Συντομότερα μονοπάτια από  $s$  προς όλες τις κορυφές:  
**Δέντρο Συντομότερων Μονοπατιών (SPT, ΔΣΜ).**
  - Αν συντομότερα  $s - v_1$  και  $s - v_2$  μονοπάτια έχουν **κοινή** κορυφή  $u$ , χρησιμοποιούν **(ίδιο)** συντομότερο  $s - u$  μονοπάτι.
  - ΔΣΜ αναπαρίσταται με **πίνακα γονέων.**



# Συντομότερα Μονοπάτια

- Ταυτίζεται ΔΣΜ με ΕΣΔ;
- Έστω συντομότερα μονοπάτια από  $s$  σε  $G(V, E, w)$ .
  - Τι συμβαίνει σε  $G(V, E, kw), k > 0$ ;
  - Τι συμβαίνει σε  $G(V, E, kw), k < 0$ ;
  - Τι συμβαίνει σε  $G(V, E, w+k), k > 0$ ;



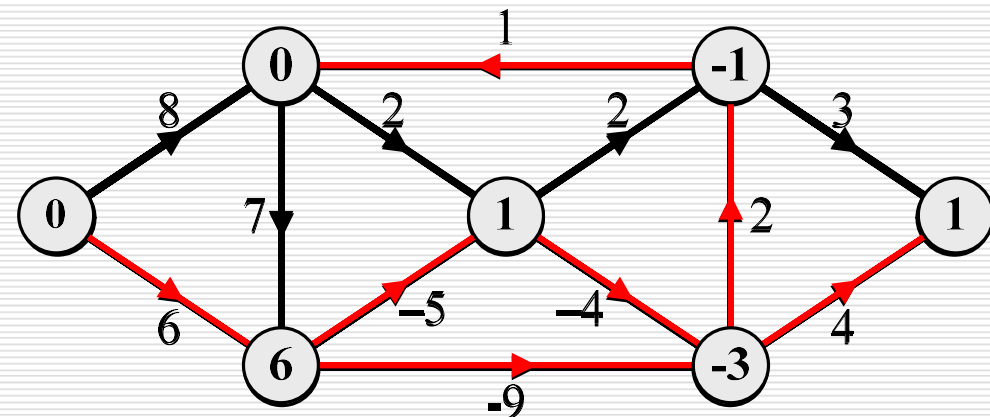
# Αποστάσεις

- Αποστάσεις ικανοποιούν την «τριγωνική ανισότητα»:

$$\forall (v, u) \in E, d(s, u) \leq d(s, v) + w(v, u)$$

$$\forall v, u \in V, d(s, u) \leq d(s, v) + d(v, u)$$

- Ισότητα ισχύει ανν συντ.  $s - u$  μονοπάτι περιέχει ακμή  $(v, u)$  (αντίστοιχα, διέρχεται από κορυφή  $v$ ).



# Υπολογισμός Συντομότερων Μονοπατιών

- Διατηρούμε «απαισιόδοξη» εκτίμηση  $D[u]$  για  $d(s, u)$ .
  - Αρχικά:  $D[s] = 0$  και  $D[u] = \infty \quad \forall u \in V \setminus \{s\}$   
 $p[u] = \text{NULL} \quad \forall u \in V$
  - Αλγόριθμος εξετάζει ακμές  $(v, u)$  και αναπροσαρμόζει  $D[u]$ .  
Αν  $D[u] > D[v] + w(v, u)$ , τότε  $D[u] \leftarrow D[v] + w(v, u)$   
 $p[u] \leftarrow v$
- $D[u]$  = μήκος συντομότερου γνωστού  $s - u$  μονοπατιού.
  - Επαγωγικά: αν ισχύει πριν τελευταία εξέταση ακμής  $(v, u)$ , ισχύει και μετά αφού  $D[u] \leftarrow \min\{D[u], D[v] + w(v, u)\}$
  - Πάντα  $D[u] \geq d(s, u)$ , και  $D[u] = \infty$  αν  $\nexists s - u$  μονοπάτι.
  - Όταν ακμές συντομότερου  $s - v$  μονοπατ. εξεταστούν με τη σειρά, γίνεται  $D[u] = d(s, u)$  και δεν μειώνεται στο μέλλον.
- Συστηματική εξέταση ακμών και κριτήριο τερματισμού.

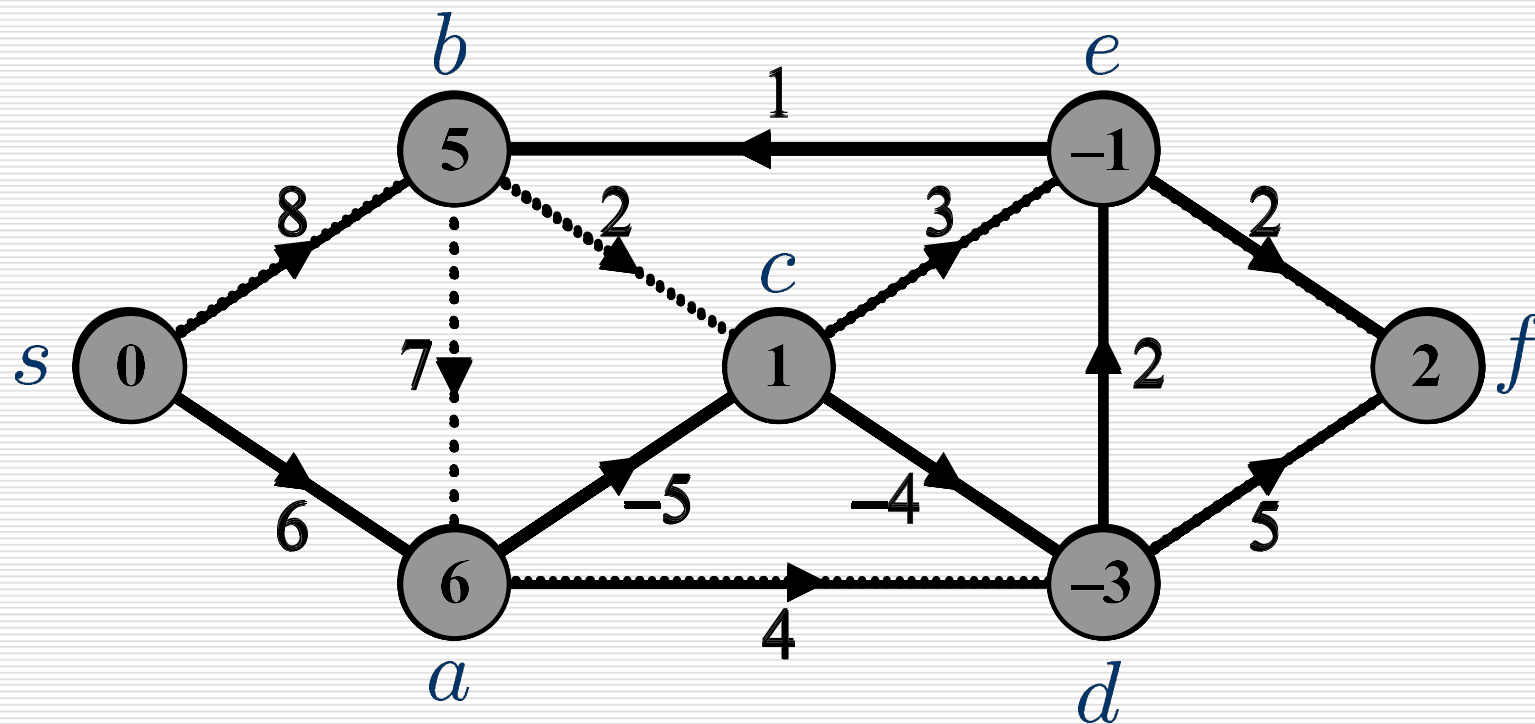
# Αλγόριθμος Bellman-Ford

- «Απαισιόδοξη» εκτίμηση  $D[u]$ .
  - Τέλος κάθε φάσης  $i$ ,  $D[u] \leq D[u, i]$
- Σε φάση  $i = 1, \dots, n-1$ , κάθε ακμή εξετάζεται μία φορά.
- Επιπλέον φάση για έλεγχο ύπαρξης κύκλου αρνητικού μήκος.
- Χρόνος εκτέλεσης  $\Theta(nm)$ .

```
Bellman-Ford( $G(V, E, w), s$ )
  for all  $u \in V$  do
     $D[u] \leftarrow \infty$ ;  $p[u] \leftarrow \text{NULL}$ ;
   $D[s] \leftarrow 0$ ;
  for  $i \leftarrow 1$  to  $n - 1$  do
    for all  $(v, u) \in E$  do
      if  $D[u] > D[v] + w(v, u)$  then
         $D[u] \leftarrow D[v] + w(v, u)$ ;
         $p[u] \leftarrow v$ ;
  for all  $(v, u) \in E$  do
    if  $D[u] > D[v] + w(v, u)$  then
      return(NEG-CYCLE);
```

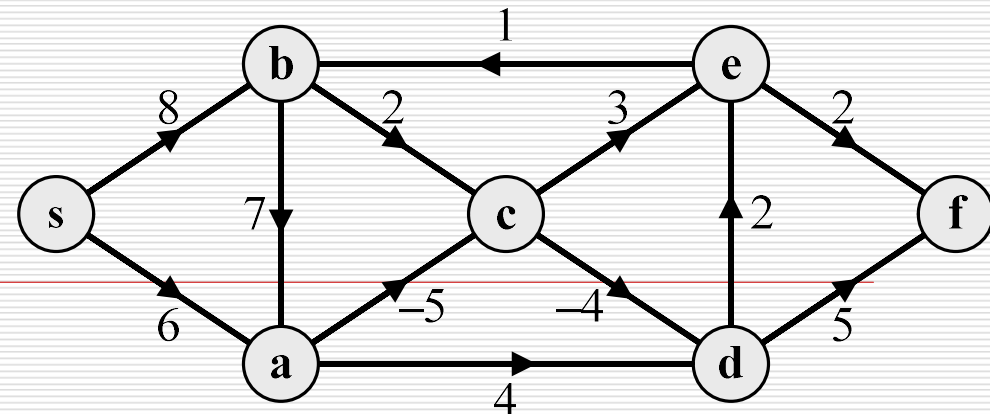


# Αλγ. Bellman-Ford: Παράδειγμα



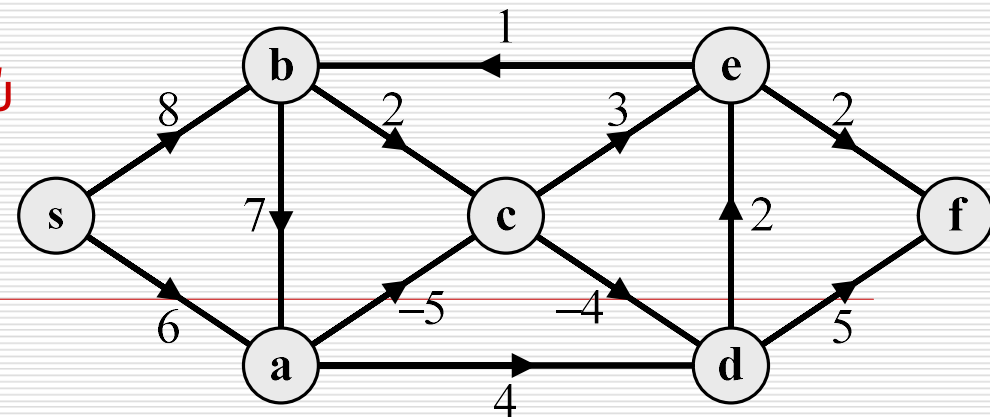
# Αλγ. Bellman-Ford ως DP

- Ιδέα: δοκιμή **όλων των ακμών** σε **κάθε πιθανή θέση** για συντομότερο  $s - u$  μονοπάτι (ταυτόχρονα για όλες τις  $u$ ).
  - $D(u, i) =$  μήκος συντομότερου  $s - u$  μονοπ. με  $\leq i$  ακμές.
  - Αρχικά  $D(s, 0) = 0$  και  $D(u, 0) = \infty$  για κάθε  $u \neq s$ .
  - Από ΣΜ με  $\leq i$  ακμές σε ΣΜ με  $\leq i+1$  ακμές:  
$$D(u, i + 1) = \min\{D(u, i), \min_{v:(v,u) \in E} \{D(v, i) + w(v, u)\}\}$$
  - (Απλό) μονοπάτι έχει  $\leq n - 1$  ακμές  $\Rightarrow D(u, n-1) = d(s, u)$   
 $D(u, n) < D(u, n-1)$  ανν κύκλος αρνητικού μήκους.
  - Υπολογισμός τιμών  $D(u, i)$ ,  
 $u \in V, i = 1, \dots, n$ , με  
**δυναμικό προγραμματισμό.**



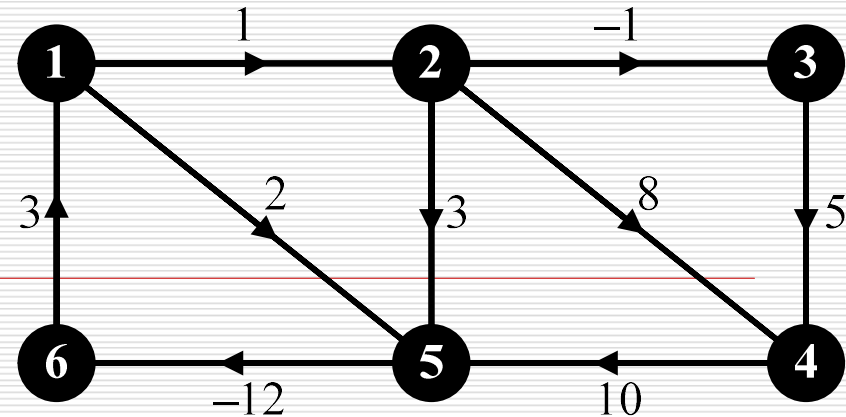
# Αλγ. Bellman-Ford: Ορθότητα

- Αν **όχι κύκλος αρνητικού μήκους**,  $D[u] = d(s, u)$  στο τέλος.
  - **Συντομότερο  $s - u$  μονοπάτι**  $s = v_0, v_1, \dots, v_k = u$  με  $k$  ακμές.
  - **Επαγωγική υπόθ.:** Τέλος φάσης  $i-1$ ,  $D[v_{i-1}] = d(s, v_{i-1})$ .
  - Τέλος φάσης  $i$ : εξέταση ακμής  $(v_{i-1}, v_i)$  και  $D[v_i] = d(s, v_i)$ :
$$d(s, v_i) \leq D[v_i] \leq D[v_{i-1}] + w(v_{i-1}, v_i)$$
$$= d(s, v_{i-1}) + w(v_{i-1}, v_i) = d(s, v_i)$$
- Τέλος φάσης  $n - 1$ :  $D[u] = d(s, u)$  για κάθε κορυφή  $u$ .
- $D[u]$  **δεν μειώνεται άλλο**, αφού πάντα  $D[u] \geq d(s, u)$ .
- Αλγόριθμος **δεν επιστρέφει ένδειξη για κύκλο αρνητικού μήκους**.



# Αλγ. Bellman-Ford: Ορθότητα

- Αν κύκλος αρνητικού μήκους, ένδειξη στο τέλος.
  - Έστω κύκλος αρνητικού μήκους  $v_0, v_1, \dots, v_{k-1}, v_k (= v_0)$  προσπελάσιμος από  $s$ .
  - Εκτιμήσεις  $D[v_i]$  πεπερασμένες στο τέλος φάσης  $n-1$ .
  - Αν όχι ένδειξη, πρέπει στη φάση  $n$  για κάθε  $v_i$  στον κύκλο:  $D[v_i] \leq D[v_{i-1}] + w(v_{i-1}, v_i)$
  - Αθροίζοντας κατά μέλη:
$$\sum_{i=1}^k D[v_i] \leq \sum_{i=1}^k D[v_{i-1}] + \sum_{i=1}^k w(v_{i-1}, v_i) \Rightarrow \sum_{i=1}^k w(v_{i-1}, v_i) \geq 0$$
  - Άτοπο! Άρα ο αλγόριθμος επιστρέφει ένδειξη για κύκλο αρνητικού μήκους.



# Συντομότερα Μονοπάτια σε DAG

- Σε DAG, σειρά εμφάνισης κορυφών σε κάθε μονοπάτι (άρα και ΔΣΜ) ακολουθεί τοπολογική διάταξη!
  - Έστω τοπολογική διάταξη  $s = v_1, v_2, \dots, v_n$ .
  - $d(s, v_k)$  εξαρτάται μόνο από  $d(s, v_j)$  με  $j < k$  :
$$d(s, v_k) = \min_{v_j: (v_j, v_k) \in E} \{d(s, v_j) + w(v_j, v_k)\}$$
- Κορυφές εντάσσονται στο ΔΣΜ με σειρά τοπολογ. διάταξης και εξετάζονται εξερχόμενες ακμές τους (μια φορά κάθε ακμή!).
  - Ορθότητα με επαγωγή (παρόμοια με Bellman-Ford).
  - Επαγωγική υπόθ.: ακριβώς πριν την ένταξη του  $v_k$  στο ΔΣΜ, ισχύει ότι  $D[v_j] = d(s, v_j)$  για κάθε  $j = 0, \dots, k$ .
  - Ακριβώς πριν ένταξη  $v_{k+1}$  στο ΔΣΜ,  $D[v_{k+1}] = d(s, v_{k+1})$  αφού
$$D[v_{k+1}] = \min_{v_j: (v_j, v_{k+1}) \in E} \{D[v_j] + w(v_j, v_{k+1})\}$$

# Συντομότερα Μονοπάτια σε DAG

ShortestPath-DAG( $G(V, E, w), v_1$ )

Έστω τοπολογική διάταξη  $v_1, v_2, \dots, v_n$ ;

**for**  $j \leftarrow 1$  **to**  $n$  **do**

$D[v_j] \leftarrow \infty$ ;  $p[v_j] \leftarrow \text{NULL}$ ;

$D[v_1] \leftarrow 0$ ;

**for**  $j \leftarrow 1$  **to**  $n - 1$  **do**

**for all**  $(v_j, v_i) \in E$  **do**

**if**  $D[v_i] > D[v_j] + w(v_j, v_i)$  **then**

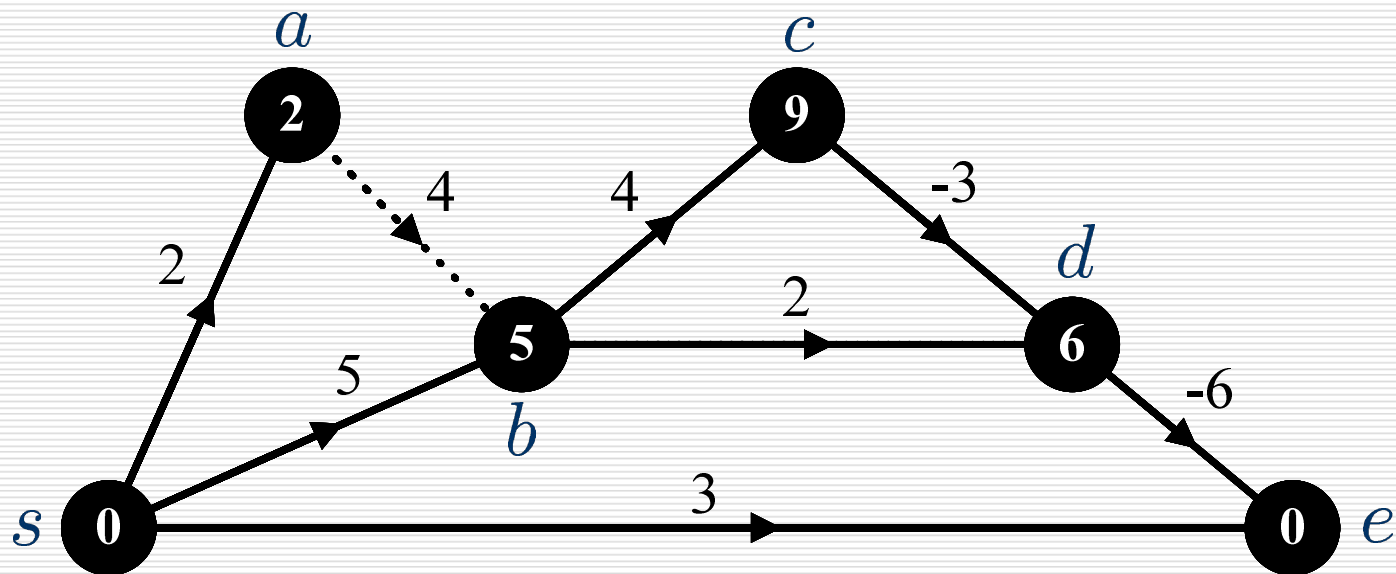
$D[v_i] \leftarrow D[v_j] + w(v_j, v_i)$ ;

$p[v_i] \leftarrow v_j$ ;

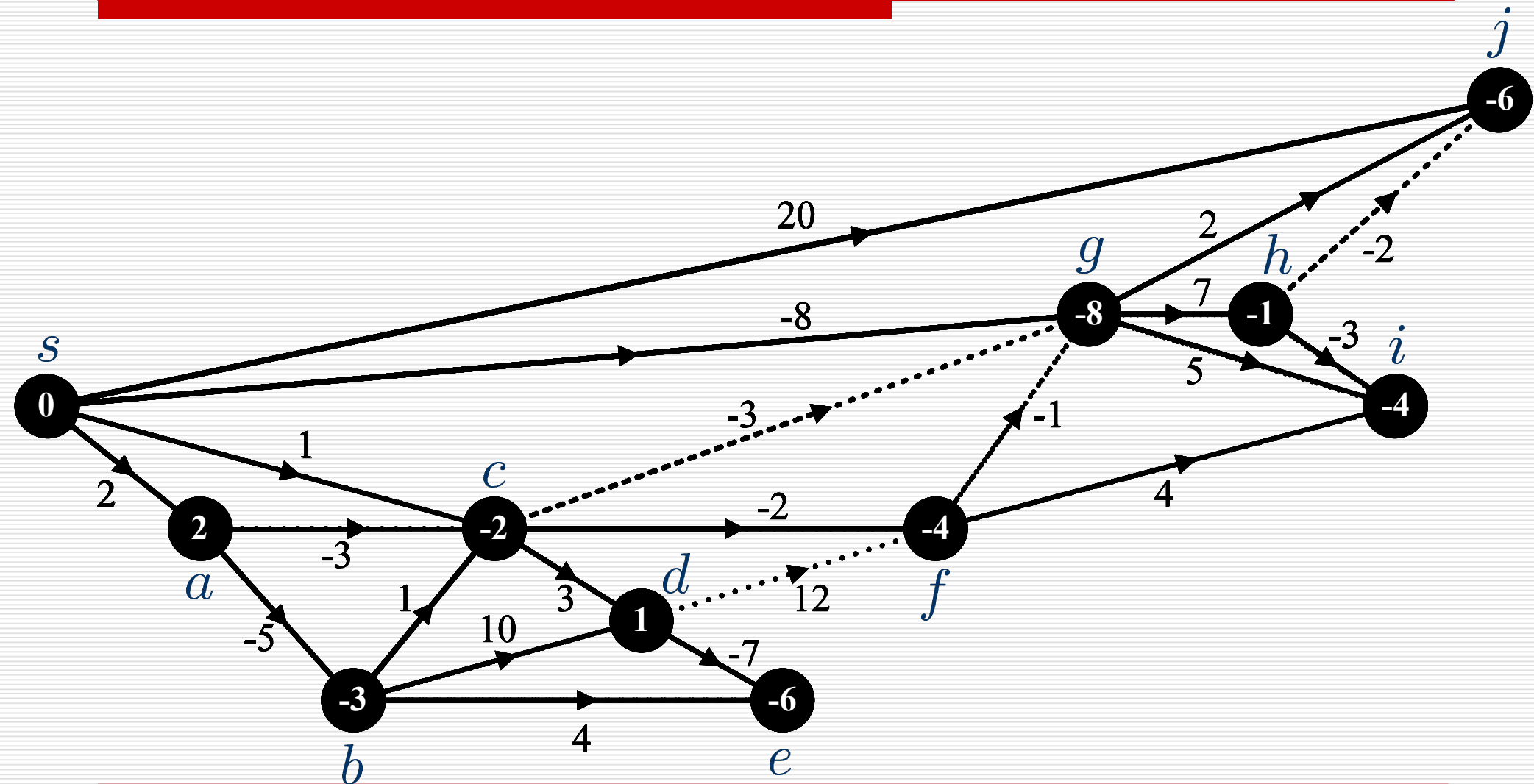
- Χρόνος εκτέλεσης: γραμμικός,  $\Theta(n+m)$
- Χρησιμοποιείται και για υπολογισμό μακρύτερων μονοπατιών.
  - Αν  $G(V, E, w)$  **ακυκλικό**,  $p$  μακρύτερο  $s - u$  μονοπάτι ανν  $p$  συντομότερη  $s - u$  διαδρομή στο  $G(V, E, -w)$ .

# Παράδειγμα

---



# Παράδειγμα



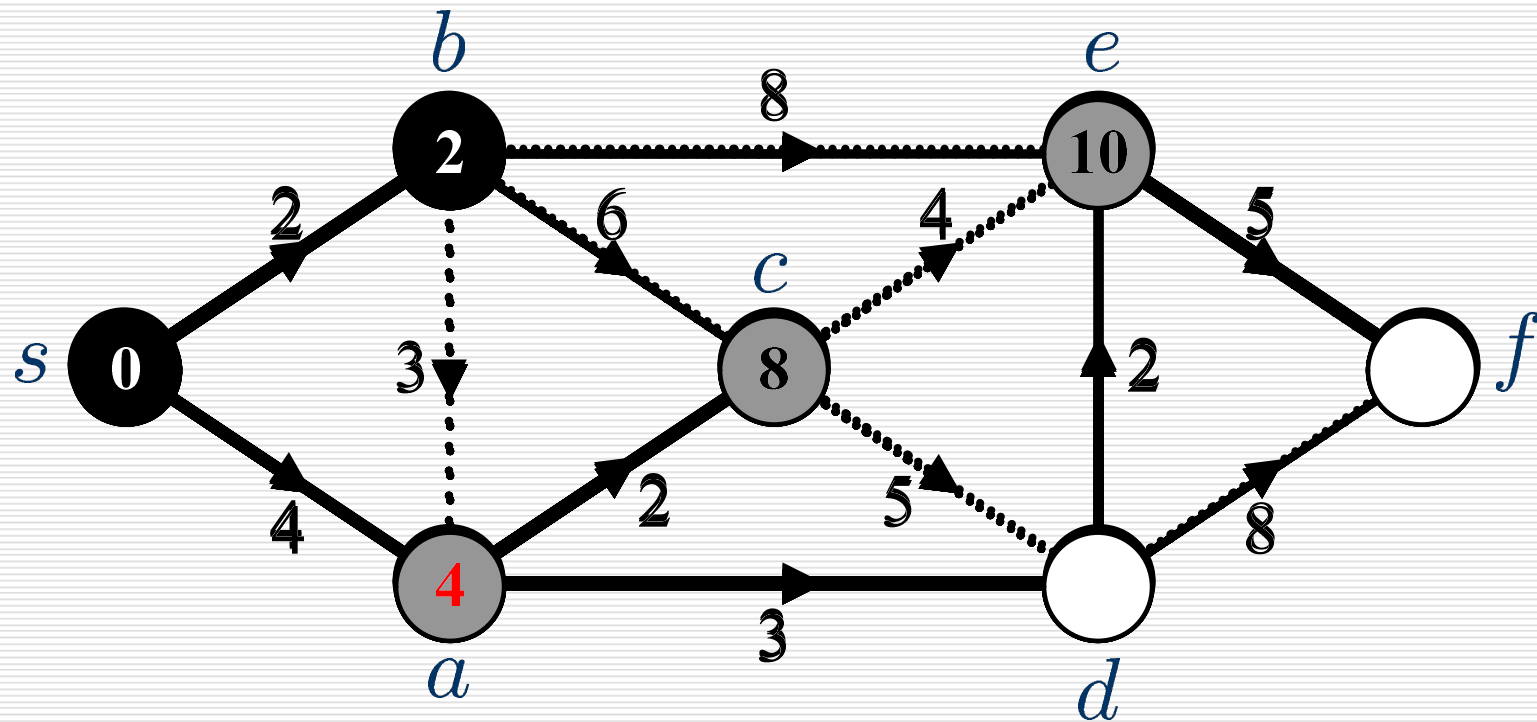


# Αλγόριθμος Dijkstra

---

- Ταχύτερα αν **όχι αρνητικά** μήκη! Αποτελεί **γενίκευση BFS**.
  - Ταχύτερα αν υπάρχει πληροφορία για **σειρά εμφάνισης κορυφών** σε **συντομότερα μονοπάτια** (και ΔΣΜ).
  - Μη αρνητικά μήκη: κορυφές σε **αύξουσα σειρά απόστασης**.
- Κορυφές εντάσσονται σε ΔΣΜ σε **αύξουσα απόσταση** και εξετάζονται **εξερχόμενες ακμές** τους (**μια φορά κάθε ακμή!**).
  - Αρχικά  $D[s] = 0$  και  $D[u] = \infty$  για κάθε  $u \neq s$ .
  - Κορυφή  $u$  εκτός ΔΣΜ με **ελάχιστο**  $D[u]$  εντάσσεται σε ΔΣΜ.
  - Για κάθε ακμή  $(u, v)$ ,  $D[v] \leftarrow \min\{D[v], D[u] + w(u, v)\}$
- **Ορθότητα**: όταν  $u$  εντάσσεται σε ΔΣΜ,  $D[u] = d(s, u)$ .
  - Μη αρνητικά μήκη: κορυφές  $v$  με **μεγαλύτερο**  $D[v]$  σε **μεγαλύτερη απόσταση** και **δεν επηρεάζουν**  $D[u]$ .

# Αλγόριθμος Dijkstra: Παράδειγμα



# Αλγόριθμος Dijkstra

□ Άπληστος αλγόριθμος.

□ **Υλοποίηση:**

- Ελάχιστο  $D[v]$ :  
ουρά προτεραιότητας.
- Binary heap:  
 $\Theta(m \log n)$
- Fibonacci heap:  
 $\Theta(m + n \log n)$
- Ελάχιστο  $D[v]$   
γραμμικά:  $\Theta(n^2)$ .

Dijkstra( $G(V, E, w), s$ )

**for all**  $u \in V$  **do**

$D[u] \leftarrow \infty$ ;  $p[u] \leftarrow \text{NULL}$ ;

$D[s] \leftarrow 0$ ;  $S \leftarrow \emptyset$ ;

**while**  $|S| < |V|$  **do**

$u \notin S : D[u] = \min_{v \notin S} \{D[v]\}$ ;

$S \leftarrow S \cup \{u\}$ ;

**for all**  $v \in \text{AdjList}[u]$  **do**

**if**  $D[v] > D[u] + w(u, v)$  **then**

$D[v] \leftarrow D[u] + w(u, v)$ ;

$p[v] \leftarrow u$ ;

# Κάτι μου Θυμίζει ...;!

---

Dijkstra( $G(V, E, w), s$ )

**for all**  $u \in V$  **do**

$D[u] \leftarrow \infty$ ;  $p[u] \leftarrow \text{NULL}$ ;

$D[s] \leftarrow 0$ ;  $S \leftarrow \emptyset$ ;

**while**  $|S| < |V|$  **do**

$u \notin S : D[u] = \min_{v \notin S} \{D[v]\}$ ;

$S \leftarrow S \cup \{u\}$ ;

**for all**  $v \in \text{AdjList}[u]$  **do**

**if**  $D[v] > D[u] + w(u, v)$  **then**

$D[v] \leftarrow D[u] + w(u, v)$ ;

$p[v] \leftarrow u$ ;

MST-Prim( $G(V, E, w), s$ )

**for all**  $u \in V$  **do**

$c[u] \leftarrow \infty$ ;  $p[u] \leftarrow \text{NULL}$ ;

$c[s] \leftarrow 0$ ;  $S \leftarrow \emptyset$ ;  $\Delta \leftarrow \emptyset$ ;

**while**  $|S| < |V|$  **do**

$u \notin S : c[u] = \min_{v \notin S} \{c[v]\}$ ;

$S \leftarrow S \cup \{u\}$ ;

**for all**  $v \in \text{AdjList}[u]$  **do**

**if**  $v \notin S$  **and**  $w(u, v) < c[v]$  **then**

$c[v] \leftarrow w(u, v)$ ;

$p[v] \leftarrow u$ ;

**if**  $p[u] \neq \text{NULL}$  **then**

$\Delta \leftarrow \Delta \cup \{u, p[u]\}$ ;

# Αλγόριθμος Dijkstra: Εξέλιξη

---

- Απόσταση (και συντομότερο μονοπάτι) από  $s$  προς **κοντινότερη** (στην  $s$ ), **2<sup>η</sup> κοντινότερη** (στην  $s$ ) κορυφή, κοκ.
- **Συντομότερα μονοπάτια** για κοντινότερες κορυφές, με υπολογισμένες αποστάσεις, σχηματίζουν **υποδέντρο του ΔΣΜ**.
- Επόμενη κοντινότερη (στην  $s$ ) κορυφή είναι **συνοριακή** κορυφή.
  - **Συνοριακή κορυφή**: **δεν ανήκει** σε υποδέντρο ΔΣΜ και έχει **εισερχόμενη ακμή** από υποδέντρο.
- Εκτιμήσεις απόστασης συνοριακών κορυφών διατηρούνται σε **ουρά προτεραιότητας**.
- Συνοριακή κορυφή με **ελάχιστη εκτίμηση απόστασης** «βγαίνει» από ουρά προτεραιότητας και **προστίθεται** στο υποδέντρο.
  - **Εκτιμήσεις απόστασης** συνοριακών κορυφών ενημερώνονται με προσθήκη νέας κορυφής στο υποδέντρο (για εξερχόμενες ακμές της).

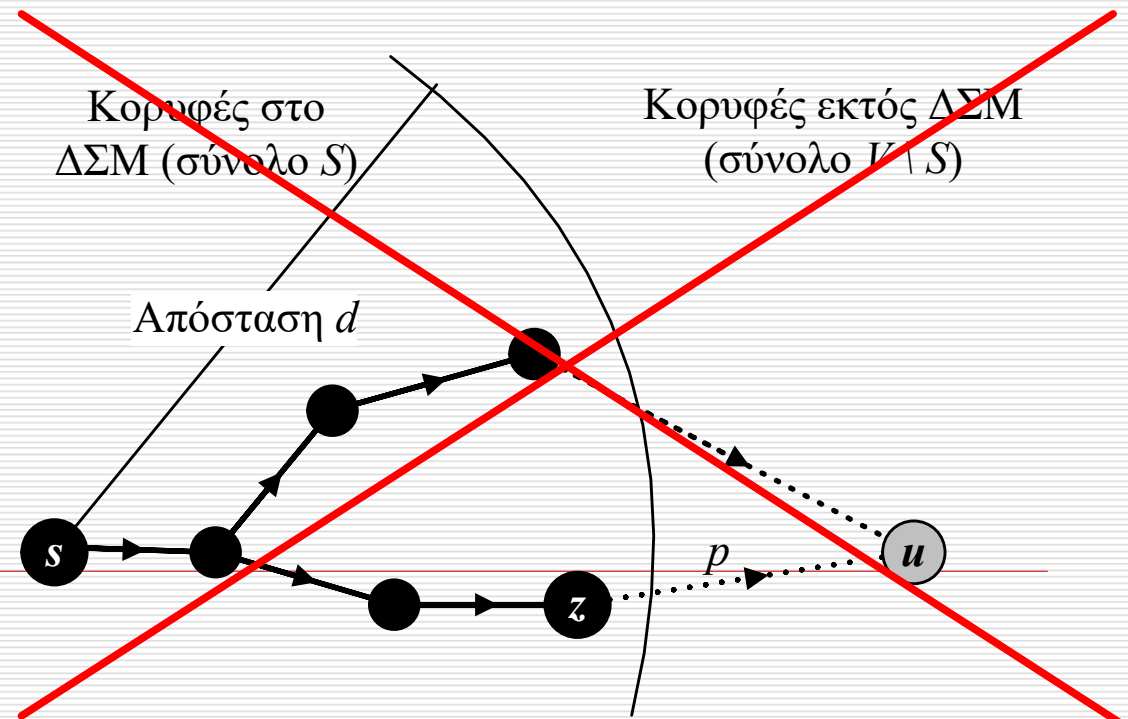
# Αλγόριθμος Dijkstra: Ορθότητα

- Θ.δ.ο **όταν** κορυφή  $u$  εντάσσεται σε ΔΣΜ,  $D[u] = d(s, u)$ .
  - **Επαγωγή:** έστω  $D[v] = d(s, v)$  για κάθε  $v$  ήδη στο ΔΣΜ.
  - $u$  έχει **ελάχιστο**  $D[u]$  (εκτός ΔΣΜ). Έστω ότι  $D[u] > d(s, u)$ .
  - $p$  συντομότερο  $s - u$  μονοπάτι με μήκος  $d(s, u) < D[u]$ , και  $z$  τελευταία κορυφή πριν  $u$  στο  $p$ :

Μπορεί  $z$  στο ΔΣΜ;

$$d(s, u) = d(s, z) + w(z, u) < D[u]$$
$$\Rightarrow z \notin S$$

**Όχι!**



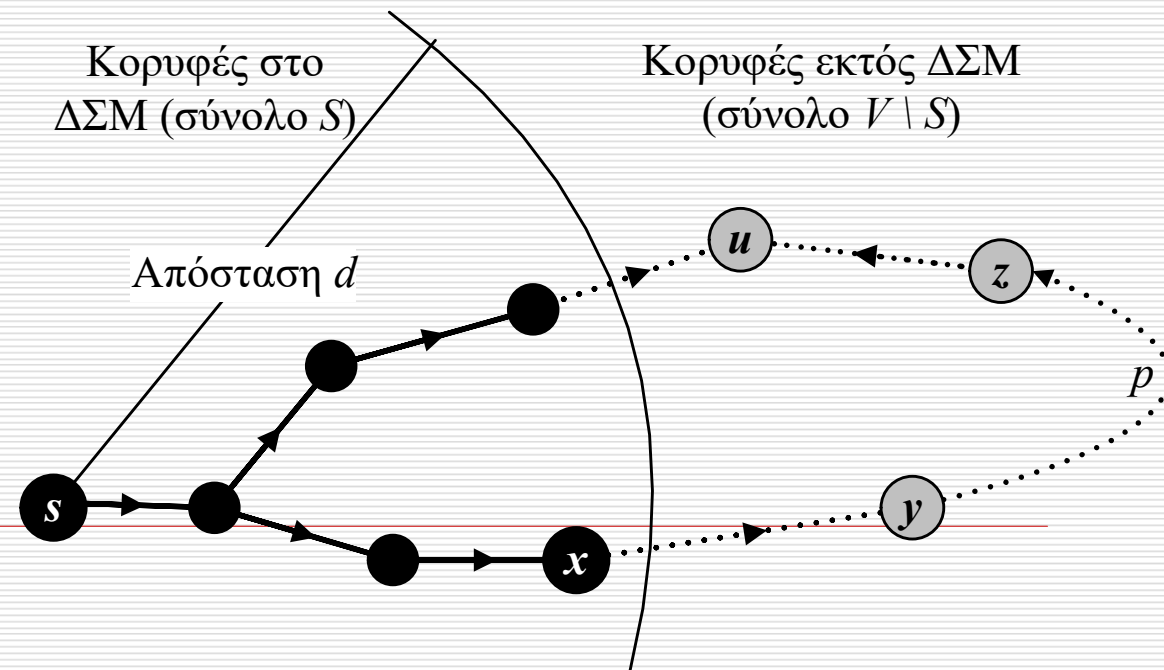
# Αλγόριθμος Dijkstra: Ορθότητα

- Θ.δ.ο **όταν** κορυφή  $u$  εντάσσεται σε ΔΣΜ,  $D[u] = d(s, u)$ .
  - **Επαγωγή:** έστω  $D[v] = d(s, v)$  για κάθε  $v$  ήδη στο ΔΣΜ.
  - $u$  έχει **ελάχιστο**  $D[u]$  (εκτός ΔΣΜ). Έστω ότι  $D[u] > d(s, u)$ .
  - $p$  συντομότερο  $s - u$  μονοπάτι με **μήκος**  $d(s, u) < D[u]$ , και  $z$  **τελευταία** κορυφή πριν  $u$  στο  $p$ :

Έστω  $x$  ( $\neq z$ ) **τελευταία** κορυφή του  $p$  στο ΔΣΜ και  $y$  (μπορεί  $y = z$ ) **επόμενη** της  $x$  στο  $p$ .

$$\begin{aligned} D[y] &\leq D[x] + w(x, y) \\ &= d(s, x) + w(x, y) \\ &= d(s, y) < D[u] \end{aligned}$$

$\Rightarrow D[y] < D[u]$ , **άτοπο!**



# Dijkstra vs Bellman-Ford

---

- Αλγ. **Dijkstra** ταχύτερος κατά  $n$  αλλά **δεν** εφαρμόζεται για **αρνητικά** μήκη.
  - Βασίζεται στο ότι **αποστάσεις δεν μειώνονται** κατά μήκος συντομότερου μονοπατιού.
- Αλγ. **Bellman-Ford** εφαρμόζεται **για αρνητικά** μήκη.
  - Αποστάσεις **μπορεί να μειώνονται** κατά μήκος συντομότερου μονοπατιού.
  - «Τελευταία» κορυφή μπορεί σε μικρότερη απόσταση από αρχική.



# Ερωτήσεις – Ασκήσεις

---

- Αρνητικά μήκη → προσθέτουμε μεγάλο αριθμό →  
→ θετικά μήκη → αλγόριθμος Dijkstra;
- Νδο BFS υπολογίζει ΔΣΜ όταν ακμές μοναδιαίου μήκους.
- Όταν μη-αρνητικά μήκη, μπορεί ένα ΔΣΜ και ένα ΕΣΔ να μην έχουν **καμία κοινή ακμή**;
- **Bottleneck** Shortest Paths:
  - Κόστος μονοπατιού  $p$ :  $c(p) = \max_{e \in p} \{w(e)\}$
  - Υπολογισμός ΔΣΜ για **bottleneck** κόστος;
  - Τροποποίηση Dijkstra λύνει Bottleneck Shortest Paths (ακόμη και για αρνητικά μήκη):  
$$\forall (v, u) \in E, D[u] \leftarrow \min\{D[u], \max\{D[v], w(v, u)\}\}$$

# Συντομότερα Μονοπάτια για Όλα τα Ζεύγη Κορυφών

- Υπολογισμός απόστασης  $d(v, u)$  και συντομότερου  $v - u$  μονοπατιού για κάθε ζεύγος  $(v, u) \in V \times V$ .
- Αλγόριθμος για ΣΜ από μία κορυφή για κάθε  $s \in V$ .
  - Αρνητικά μήκη: Bellman-Ford σε χρόνο  $\Theta(n^2 m)$ .
  - Μη-αρνητικά μήκη: Dijkstra σε χρόνο  $\Theta(n m + n^2 \log n)$ .
- Αρνητικά μήκη: Floyd-Warshall σε χρόνο  $\Theta(n^3)$ .
- Αναπαράσταση λύσης:
  - Αποστάσεις: πίνακας  $D[1..n][1..n]$
  - Συντομότερα μονοπάτια:  $n$  ΔΣΜ, ένα για κάθε αρχική κορυφή.
    - Πίνακας  $P[1..n][1..n]$ :  $n$  πίνακες γονέων.
    - Γραμμή  $P[i]$ : πίνακας γονέων  $\Delta\Sigma\text{M}(v_i)$ .

# Αλγόριθμος Floyd-Warshall

- Θεωρούμε **γράφημα**  $G(V, E, w)$  με μήκη στις ακμές.
  - Καθορισμένη (αυθαίρετη) **αρίθμηση** κορυφών  $v_1, v_2, \dots, v_n$ .
- Αναπαράσταση γραφήματος με **πίνακα γειτνίασης**:

$$w(v_i, v_j) = \begin{cases} 0 & v_i = v_j \\ w(v_i, v_j) & v_i \neq v_j \text{ } (v_i, v_j) \in E \\ \infty & v_i \neq v_j \text{ } (v_i, v_j) \notin E \end{cases}$$

- Υπολογισμός απόστασης  $d(v_i, v_j)$  από  $d(v_i, v_k), d(v_k, v_j)$  για **όλα τα**  $k \in V \setminus \{v_i, v_j\}$ :

$$d(v_i, v_j) = \min\{w(v_i, v_j), \min_{v_k \in V \setminus \{v_i, v_j\}} \{d(v_i, v_k) + d(v_k, v_j)\}\}$$

- Φαύλος κύκλος( $;$ ):  $d(v_i, v_k) \rightarrow d(v_i, v_j)$  και  $d(v_i, v_j) \rightarrow d(v_i, v_k)$
- **Δυναμικός προγραμματισμός**: υπολογισμός **όλων** με συστηματικό **bottom-up** τρόπο!

# Αλγόριθμος Floyd-Warshall

- $D_k[v_i, v_j]$ : μήκος συντομότερου  $v_i - v_j$  μονοπατιού με ενδιάμεσες κορυφές μόνο από  $V_k = \{v_1, \dots, v_k\}$ 
  - Αρχικά  $D_0[v_i, v_j] = w(v_i, v_j)$  γιατί  $V_0 = \emptyset$ .
  - Έστω ότι γνωρίζουμε  $D_{k-1}[v_i, v_j]$  για όλα τα ζεύγη  $v_i, v_j$ .
  - $D_k[v_i, v_j]$  διέρχεται από  $v_k$  καμία ή μία φορά (μονοπάτι!):
$$D_k[v_i, v_j] = \min\{D_{k-1}[v_i, v_j], D_{k-1}[v_i, v_k] + D_{k-1}[v_k, v_j]\}$$

- Αναδρομική σχέση για  $D_0, D_1, \dots, D_n$ :

$$D_k[v_i, v_j] = \begin{cases} w(v_i, v_j) & k = 0 \\ \min\{D_{k-1}[v_i, v_j], D_{k-1}[v_i, v_k] + D_{k-1}[v_k, v_j]\} & k = 1, \dots, n \end{cases}$$

- Υπολογισμός  $D_n$  με **δυναμικό προγραμματισμό**.
- Κύκλος αρνητικού μήκους αν  $D_n[v_i, v_i] < 0$ .

# Αλγόριθμος Floyd-Warshall

- Τυπικός δυναμικός προγραμματισμός:

Χρόνος:  $\Theta(n^3)$

Floyd-Warshall( $G(V, E, w)$ )

**for**  $i \leftarrow 1$  **to**  $n$  **do**

**for**  $j \leftarrow 1$  **to**  $n$  **do**

**if**  $(v_i, v_j) \in E$  **then**  $D_0[i, j] \leftarrow w(v_i, v_j)$ ;

**else**  $D_0[i, j] \leftarrow \infty$ ;

$D_0[i, i] \leftarrow 0$ ;

**for**  $k \leftarrow 1$  **to**  $n$  **do**

**for**  $i \leftarrow 1$  **to**  $n$  **do**

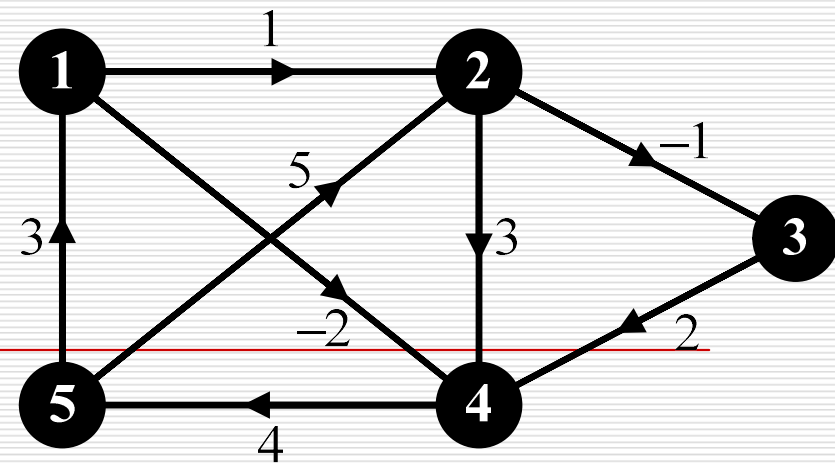
**for**  $j \leftarrow 1$  **to**  $n$  **do**

**if**  $D_{k-1}[i, j] > D_{k-1}[i, k] + D_{k-1}[k, j]$  **then**

$D_k[i, j] \leftarrow D_{k-1}[i, k] + D_{k-1}[k, j]$ ;

**else**  $D_k[i, j] \leftarrow D_{k-1}[i, j]$ ;

# Παράδειγμα



$$D_0 = \begin{pmatrix} 0 & 1 & \infty & -2 & \infty \\ \infty & 0 & -1 & 3 & \infty \\ \infty & \infty & 0 & 2 & \infty \\ \infty & \infty & \infty & 0 & 4 \\ 3 & 5 & \infty & \infty & 0 \end{pmatrix}$$

$$D_2 = \begin{pmatrix} 0 & 1 & 0 & -2 & \infty \\ \infty & 0 & -1 & 3 & \infty \\ \infty & \infty & 0 & 2 & \infty \\ \infty & \infty & \infty & 0 & 4 \\ 3 & 4 & 3 & 1 & 0 \end{pmatrix}$$

$$D_4 = \begin{pmatrix} 0 & 1 & 0 & -2 & 2 \\ \infty & 0 & -1 & 1 & 5 \\ \infty & \infty & 0 & 2 & 6 \\ \infty & \infty & \infty & 0 & 4 \\ 3 & 4 & 3 & 1 & 0 \end{pmatrix}$$

$$D_1 = \begin{pmatrix} 0 & 1 & \infty & -2 & \infty \\ \infty & 0 & -1 & 3 & \infty \\ \infty & \infty & 0 & 2 & \infty \\ \infty & \infty & \infty & 0 & 4 \\ 3 & 4 & \infty & 1 & 0 \end{pmatrix}$$

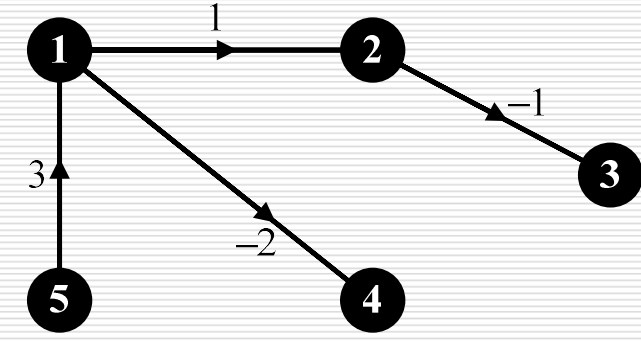
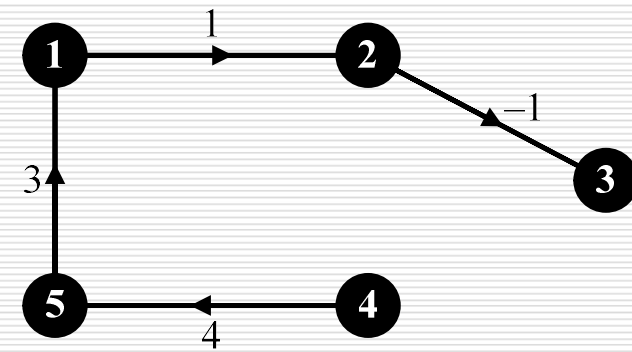
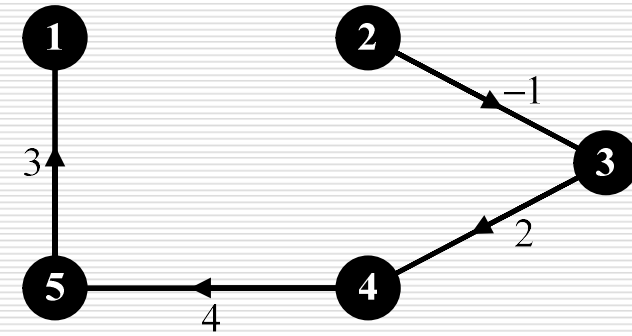
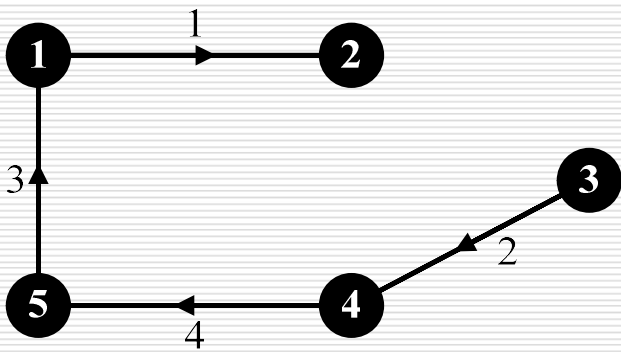
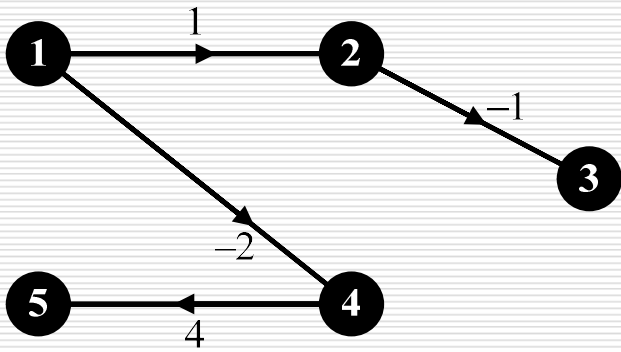
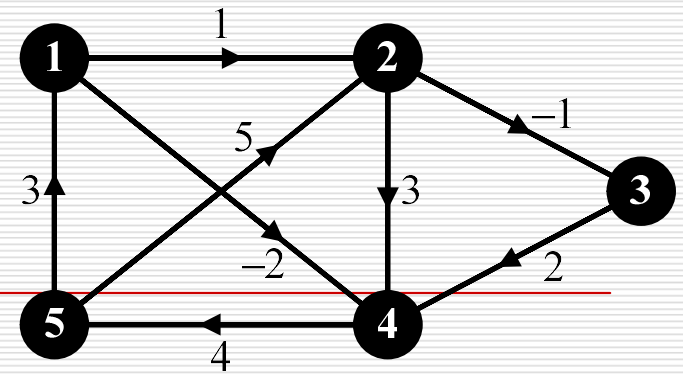
$$D_3 = \begin{pmatrix} 0 & 1 & 0 & -2 & \infty \\ \infty & 0 & -1 & 1 & \infty \\ \infty & \infty & 0 & 2 & \infty \\ \infty & \infty & \infty & 0 & 4 \\ 3 & 4 & 3 & 1 & 0 \end{pmatrix}$$

$$D_5 = \begin{pmatrix} 0 & 1 & 0 & -2 & 2 \\ 8 & 0 & -1 & 1 & 5 \\ 9 & 10 & 0 & 2 & 6 \\ 7 & 8 & 7 & 0 & 4 \\ 3 & 4 & 3 & 1 & 0 \end{pmatrix}$$

# Υπολογισμός Συντομότερων Μονοπατιών

- $P_k[v_i, \cdot]$  : ΔΣΜ( $v_i$ ) με **ενδιάμεσες** κορυφές μόνο από  $V_k$ .
  - Αποστάσεις  $D_k[v_i, \cdot]$  αντιστοιχούν σε μονοπάτια  $P_k[v_i, \cdot]$ .
  - $P_k[v_i, v_j]$ : προηγούμενη κορυφή της  $v_j$  στο **συντομότερο**  $v_i - v_j$  μονοπάτι με **ενδιάμεσες** κορυφές μόνο από  $V_k$ .
- $P_0$  καθορίζεται από πίνακα γειτνίασης: 
$$P_0[v_i, v_j] = \begin{cases} \text{NULL} & \text{αν } i = j \text{ ή } (v_i, v_j) \notin E \\ v_i & \text{διαφορετικά} \end{cases}$$
- Αναδρομική σχέση για  $P_0, P_1, \dots, P_n$  :
$$P_k[v_i, v_j] = \begin{cases} P_{k-1}[v_i, v_j] & D_{k-1}[v_i, v_j] \leq D_{k-1}[v_i, v_k] + D_{k-1}[v_k, v_j] \\ P_{k-1}[v_k, v_j] & D_{k-1}[v_i, v_j] > D_{k-1}[v_i, v_k] + D_{k-1}[v_k, v_j] \end{cases}$$
  - Υπολογισμός  $P_n$  ταυτόχρονα με υπολογισμό  $D_n$ .
  - Εύκολη τροποποίηση προηγούμενης υλοποίησης.

# Παράδειγμα



$$P_5 = \begin{pmatrix} \text{NULL} & 1 & 2 & 1 & 4 \\ 5 & \text{NULL} & 2 & 3 & 4 \\ 5 & 1 & \text{NULL} & 3 & 4 \\ 5 & 1 & 2 & \text{NULL} & 4 \\ 5 & 1 & 2 & 1 & \text{NULL} \end{pmatrix}$$



# Αλγόριθμος Johnson

---

- Συντομότερα μονοπάτια για όλα τα ζεύγη κορυφών σε αραιά γραφήματα με αρνητικά μήκη:
  - **Μετατροπή** αρνητικών μηκών σε **μη αρνητικά** χωρίς να αλλάξουν τα συντομότερα μονοπάτια.
- Αλγόριθμος για γράφημα  $G(V, E, w)$ :
  - Νέα κορυφή  $s$  που συνδέεται με κάθε  $u \in V$  με ακμή μηδενικού μήκους:  $G'(V \cup \{s\}, E \cup \{(s, u)\}, w)$ .
  - Bellman-Ford για  $G'$  με αρχική κορυφή  $s$ . Έστω  $h(u)$  απόσταση κορυφής  $u \in V$  από  $s$ .
  - Αν όχι κύκλος αρνητικού μήκους, υπολόγισε **νέα** (μη αρνητικά) μήκη:  $\hat{w}(v, u) = w(v, u) + h(v) - h(u), \forall (v, u) \in E$
  - Για κάθε  $u \in V$ , **Dijkstra** σε  $G(V, E, \hat{w})$  με αρχική κορυφή  $u$ .

# Αλγόριθμος Johnson

- Χρονική πολυπλοκότητα:
  - Bellman-Ford και  $n$  φορές Dijkstra:  $\Theta(nm + n^2 \log n)$ .
- Ορθότητα:
  - **Νέα μήκη μη αρνητικά:**  $h(\cdot)$  αποστάσεις από  $s$ , και ισχύει ότι
$$\forall (v, u) \in E, h(u) \leq h(v) + w(v, u) \Rightarrow \hat{w}(v, u) \geq 0$$
  - Μεταβολή στα μήκη **δεν επηρεάζει** συντομότερα μονοπάτια.
  - Μήκος **κάθε**  $a - \beta$  μονοπατιού μεταβάλλεται κατά  $h(\beta) - h(a)$ .
  - Έστω  $p \equiv (a = v_0, v_1, \dots, v_k = \beta)$  οποιοδήποτε  $a - \beta$  μονοπάτι.

$$\begin{aligned}\hat{\ell}(p) &= \sum_{i=0}^{k-1} \hat{w}(v_i, v_{i+1}) = \sum_{i=0}^{k-1} [w(v_i, v_{i+1}) + h(v_i) - h(v_{i+1})] \\ &= \sum_{i=0}^{k-1} w(v_i, v_{i+1}) + h(v_0) - h(v_k) = \ell(p) + h(\alpha) - h(\beta)\end{aligned}$$

# Σύνοψη

---

- Συντομότερα μονοπάτια από μία αρχική κορυφή  $s$ :
  - **Αρνητικά μήκη**: Bellman-Ford σε χρόνο  $\Theta(nm)$ .
    - Δυναμικός προγραμματισμός.
  - DAGs με αρνητικά μήκη σε χρόνο  $\Theta(m + n)$ .
  - Μη-αρνητικά μήκη: Dijkstra σε χρόνο  $\Theta(m + n \log n)$ .
    - (Προσαρμοστικός) άπληστος αλγόριθμος.
- Συντομότερα μονοπάτια για όλα τα ζεύγη κορυφών:
  - **Αρνητικά μήκη**: Floyd-Warshall σε χρόνο  $\Theta(n^3)$ .
    - Δυναμικός προγραμματισμός.
  - (Μη-)αρνητικά μήκη και **αραιά** γραφήματα,  $m = o(n^2)$ :
    - $n$  φορές Dijkstra σε χρόνο  $\Theta(nm + n^2 \log n)$ .
    - Αν αρνητικά μήκη, **αλγ. Johnson** για μετατροπή σε θετικά!