# ΕΑΠ – Θ.Ε. ΠΛΗ36
# Σύγχρονα Δίκτυα και Υπηρεσίες

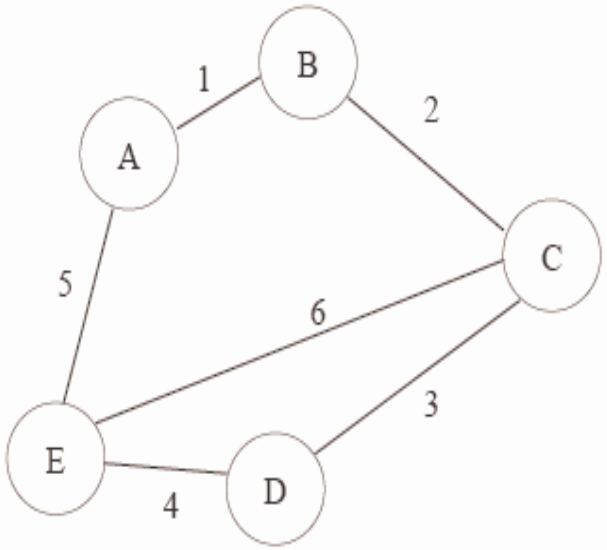
Διαφάνειες ΟΣΣ 05
Ακ. Έτος 2007-2008

Δρ. Ι. Μαριάς
Λέκτορας
Τμήμα Πληροφορικής – Ο.Π.Α.

# Περιεχόμενα

- **Εισαγωγικές Έννοιες Γράφων**
- **Βασικοί Αλγόριθμοι Γράφων**
  - Ελάχιστου Δένδρου Επικάλυψης (ΕΔΕ)
  - Συντομότερου μονοπατιού
  - Επιμερισμός Χωρητικότητας συνδέσεων σε ροές
  - Ευρεστικές μέθοδοι ΕΔΕ
- **Δρομολόγηση**
- **Ανάθεση Χωρητικότητας**

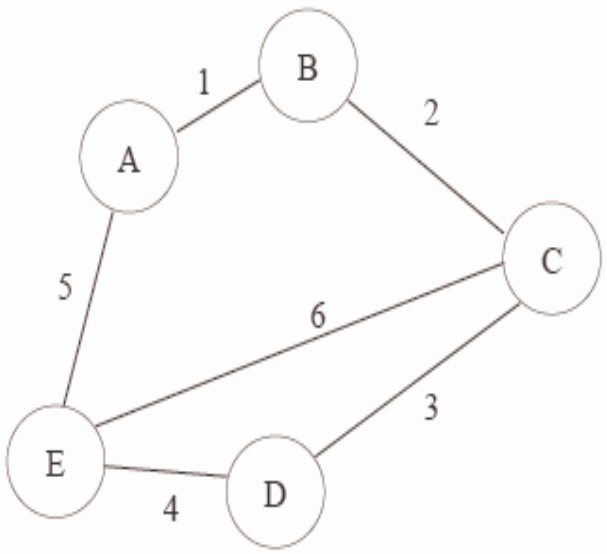
# Βασικές Έννοιες Γράφων




Σχήμα 1. Απλή δομή γράφου

- μη κατευθυντικός (undirected)
  - με συνδέσμους
- κατευθυντικός (directed)
  - με τόξα
- συνδεδεμένος
  - αν υπάρχει τουλάχιστον ένα μονοπάτι μεταξύ κάθε ζεύγους κόμβων
- δίκτυο (network)
  - εάν οι σύνδεσμοι και οι κόμβοι διαθέτουν χαρακτηριστικά
    - μήκος, χωρητικότητα, τύπος

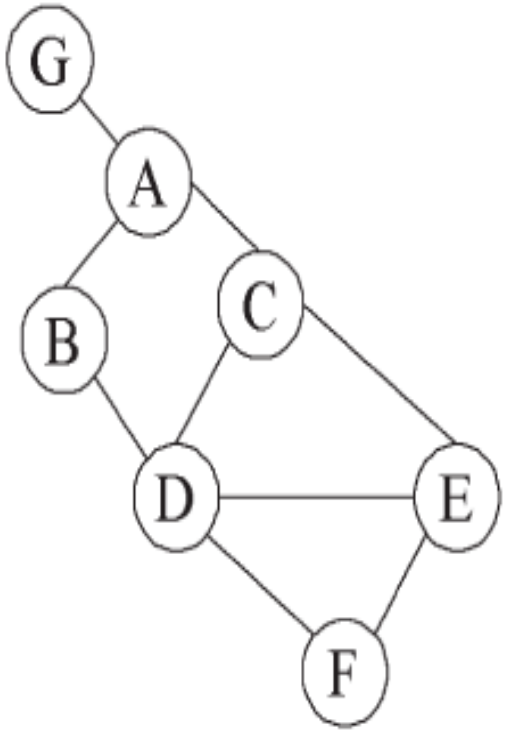
# Βασικές Έννοιες Γράφων




Σχήμα 1. Απλή δομή γράφου

- Δέντρο (tree)
  - γράφος χωρίς κυκλώματα
- δέντρο επικάλυψης (spanning tree)
  - συνδεδεμένος γράφος χωρίς κυκλώματα
  - γράφοσμε N κόμβους, το κάθε δένδρο επικάλυψης έχει N-1 ακμές
- ιδιότητες
  - Συνδεδεμένα κατ' ελάχιστο (minimally connected)
  - υπάρχει ακριβώς ένα μονοπάτι μεταξύ κάθε ζεύγους κόμβων
  - οποιαδήποτε ακμή είναι ένα ελάχιστο σύνολο αποκοπής
- δέντρο με ελάχιστο συνολικό κόστος
  - ελάχιστο δέντρο επικάλυψης (minimum spanning tree)

# Βασικές Έννοιες Γράφων

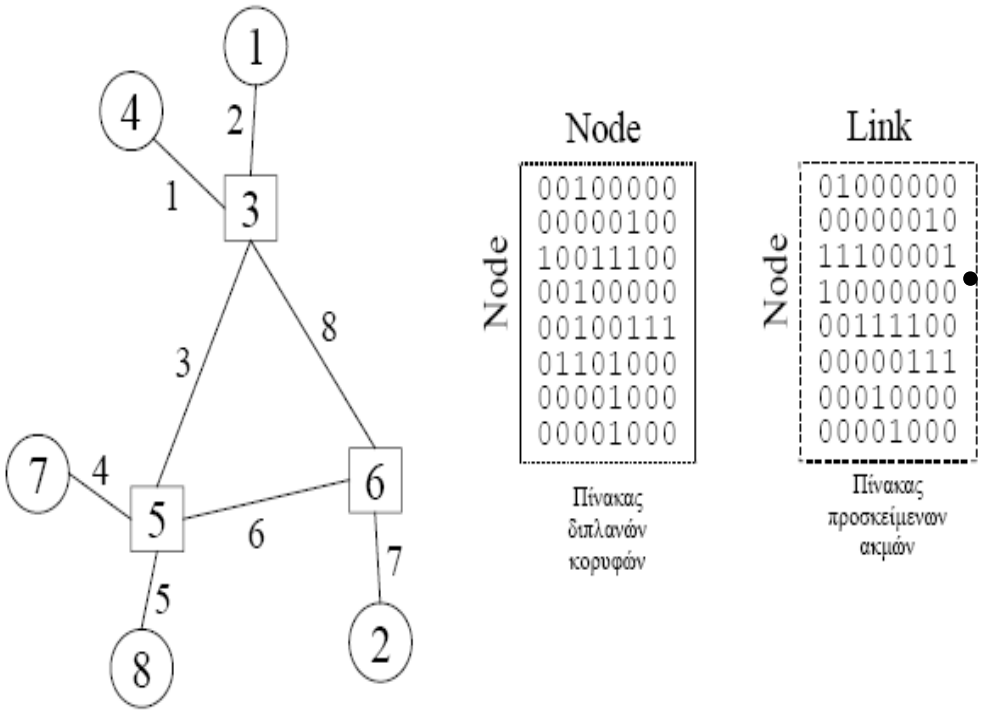



Το σύνολο των συνδέσμων: {(A,C), (B,D)} και {(C,E),(D,E),(E,F)} είναι ελάχιστα σύνολα αποκοπής.

Το σύνολο των κόμβων {C,D} αποτελεί μία τομή

- Ένα σύνολο αποσύνδεσης που τέμνει το σύνολο των κόμβων σε δύο σύνολα X και Y, καλείται σύνολο αποκοπής (cutset) X-Y.
- Ενδιαφέρουν τα ελάχιστα σύνολα αποκοπής
  - τα οποία δεν είναι υποσύνολα άλλων
  - Σε ένα δένδρο, οποιαδήποτε ακμή είναι ελάχιστο σύνολο αποκοπής.
- Ένα ελάχιστο σύνολο κόμβων των οποίων η αφαίρεση διαιρεί τους υπόλοιπους κόμβους σε δύο σύνολα, καλείται τομή (cut).

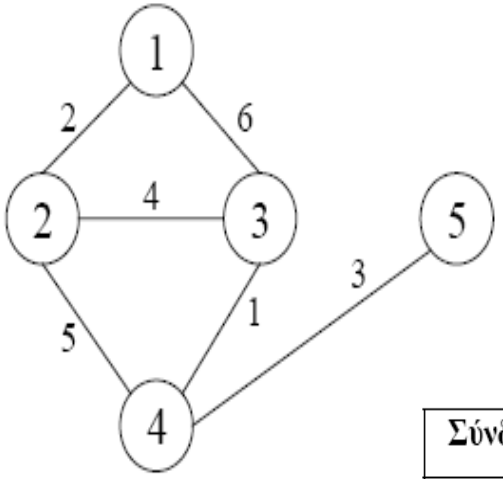
# Βασικές Έννοιες Γράφων




**Δίκτυο, πίνακας διπλανών κορυφών, προσκείμενων ακμών**

- πίνακας αραιός (sparse) εάν η πλειοψηφία των στοιχείων είναι 0.
  - όταν το πλήθος των μη μηδενικών στοιχείων είναι μία τάξη μικρότερο από το μέγεθος του πίνακα.
- Βαθμός κόμβου
  - Υπολογίζεται αθροίζοντας τον αριθμό των 1 στην στήλη που αντιστοιχεί στο συγκεκριμένο κόμβο στον πίνακα διπλανών κορυφών ή προσκείμενων ακμών.
  - Σε κατευθυντικά δίκτυα, ο **έσω-βαθμός (in degree) ή έξω-βαθμός (out degree)** μπορεί να υπολογιστεί αθροίζοντας το πλήθος των +1 ή -1 αντίστοιχα.

# Βασικές Έννοιες Γράφων




**Αναπαράσταση Δικτύου με γράφο**

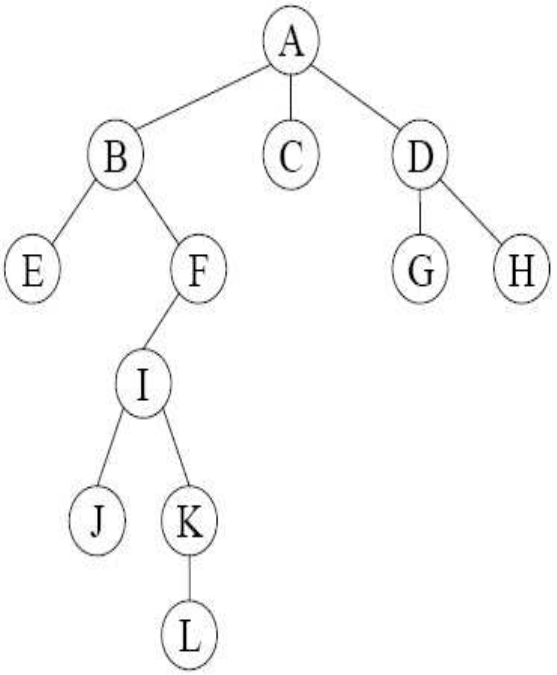
| Σύνδεσμος | Τερματισμός 1 | Τερματισμός 2 | Επόμενη ακμή 1 | Επόμενη ακμή 2 |
|---|---|---|---|---|
| 1 | 3 | 4 | 4 | 3 |
| 2 | 2 | 1 | 4 | 6 |
| 3 | 4 | 5 | 5 | 0 |
| 4 | 2 | 3 | 5 | 6 |
| 5 | 4 | 2 | 0 | 0 |
| 6 | 1 | 3 | 0 | 0 |

# Βασικοί Αλγόριθμοι Γράφων

- Εντοπισμός Δέντρων σε Γράφους
  - Πως εντοπίζω δένδρα σε ένα γράφο;
  - Γιατί πρέπει να εντοπίσω δένδρα;
    - Αποτελούν ελάχιστες (minimal) μορφές δικτύων
      - Παρέχουν συνδεσιμότητα χωρίς πλεονάζοντες συνδέσμους.
    - Παρέχοντας ένα και μοναδικό μονοπάτι μεταξύ κάθε ζεύγους κόμβων, εξαλείφουν την ανάγκη δρομολόγησης
  - Ο σχεδιασμός όμως ενός δικτύου, σχεδόν πάντα, ξεκινάει με ένα δέντρο

# Βασικοί Αλγόριθμοι Γράφων

- Εντοπισμός Δέντρων σε Γράφους
  - **Μέθοδοι διάσχισης:** Δοθέντος ενός δέντρου είναι συχνά αναγκαία η επίσκεψη όλων των κόμβων του




- αναζήτηση με προτεραιότητα πλάτους (Breadth First Search).
  - επισκέπτεται πρώτα τους κόμβους που βρίσκονται κοντά στη ρίζα
- αναζήτηση με προτεραιότητα βάθους (Depth First Search)
  - η σειρά επίσκεψης είναι η ακόλουθη: A, B, E, F, I, J, K, L, C, D, G, H

# Βασικοί Αλγόριθμοι Γράφων

- **Εντοπισμός Ελάχιστων Δέντρων σε Γράφους**
  - **Άπληστος Αλγόριθμος:** σε κάθε βήμα του επιλέγουμε την ακμή με το μικρότερο μήκος
  - Μυωπικός
  - Αλγόριθμοι Prim και Kruskal
- Αλγόριθμος Kruskal
  - ταξινομεί τις ακμές του γράφου,
  - επιλέγει αυτήν με το μικρότερο βάρος
  - συμπεριλαμβάνει τις ακμές που δεν έχουν ήδη επιλεγεί και δεν διαμορφώνουν κύκλο με στις υπόλοιπες ακμές του ελάχιστου δέντρου επικάλυψης

# Βασικοί Αλγόριθμοι Γράφων

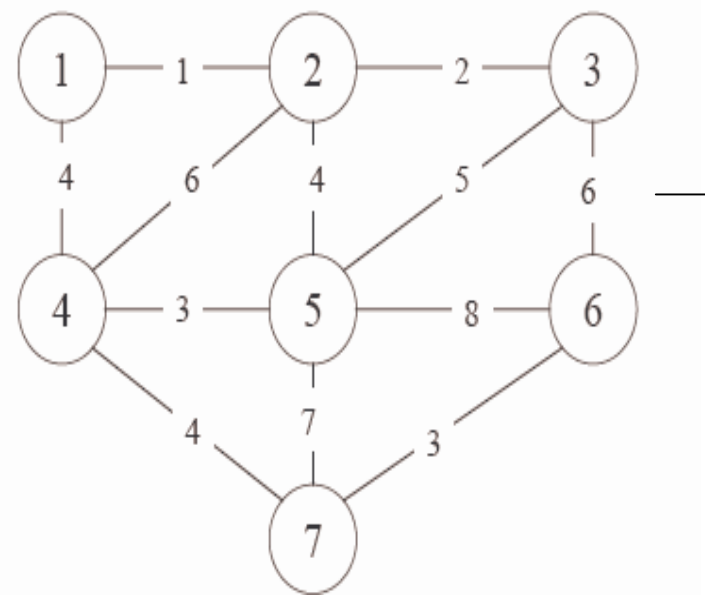
- **Εντοπισμός Ελάχιστων Δέντρων σε Γράφους**
- Αλγόριθμος Kruskal
  1. Αρχικά ορίζει ένα κενό σύνολο ακμών Τ
  2. Ορίζει ένα σύνολο συστατικών C (ένα δάσος από δέντρα)
  3. Εξετάζει όλες τις ακμές στο σύνολο των ακμών του γράφου (Ε) ταξινομώντας αυτές σύμφωνα με το βάρος τους

     δηλαδή η ακμή με το μικρότερο βάρος πρώτη
  4. Εάν μία ακμή συνδέει δύο ασύνδετα συστατικά του C, τότε προστίθεται στο Τ.

     Εάν μία ακμή δεν συνδέει δύο ασύνδετα συστατικά του C τότε απορρίπτεται.
  5. Τα βήματα 3 και 4 επαναλαμβάνονται μέχρι το C να περιέχει μόνο ένα συστατικό

# Βασικοί Αλγόριθμοι Γράφων

- **Εντοπισμός Ελάχιστων Δέντρων σε Γράφους**
- Αλγόριθμος Kruskal

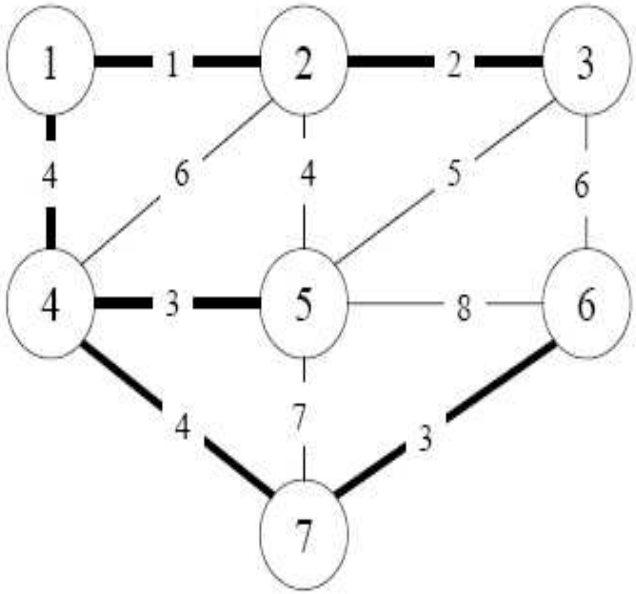



**3ο βήμα, ταξινόμηση των ακμών σύμφωνα με το βάρος**

| Ακμή | Βάρος |
|---|---|
| (1,2) | 1 |
| (2,3) | 2 |
| (4,5) | 3 |
| (6,7) | 3 |
| (1,4) | 4 |
| (2,5) | 4 |
| (4,7) | 4 |
| (3,5) | 5 |
| (2,6) | 6 |
| (3,6) | 6 |
| (5,7) | 7 |
| (5,6) | 8 |

# Βασικοί Αλγόριθμοι Γράφων

- **Εντοπισμός Ελάχιστων Δέντρων σε Γράφους**
- Αλγόριθμος Kruskal




**4ο βήμα, στο Τ προστίθενται οι ακμές χαμηλότερου βάρους οι οποίες όμως δεν προκαλούν την εμφάνιση κυκλώματος.**

**Με επανάληψη**

**Το δέντρο Τ διαμορφώνεται όπως φαίνεται δίπλα με τη σειρά προσθήκης ακμών.**

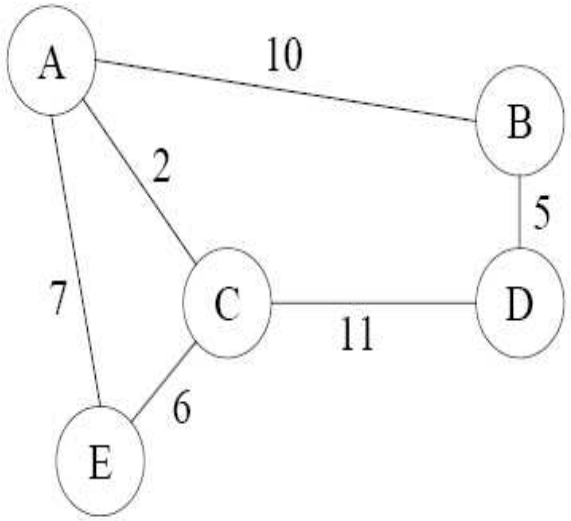
**(1,2) → (2,3) → (4,5) → (6,7) → (1,4) → (4,7)**

# Βασικοί Αλγόριθμοι Γράφων

- **Εντοπισμός Ελάχιστων Δέντρων σε Γράφους**
- Αλγόριθμος Prim. ξεκινάει από μία κορυφή και αναπτύσσει το υπόλοιπο του δέντρου με την προσθήκη μίας ακμής σε κάθε βήμα
  - Διαμορφώνει ένα δέντρο με μοναδική κορυφή
    - επιλέγεται τυχαία από τον γράφο.
  - Δημιουργεί ένα σύνολο που περιέχει όλες τις ακμές του γράφου
  - Επαναλαμβάνει (loop) τις παρακάτω ενέργειες όσο υπάρχουν κορυφές εκτός δέντρου
    - ή η κάθε ακμή στο άνω σύνολο να συνδέει δύο ακμές του δέντρου
  - Αφαίρεση μίας ακμής ελαχίστου κόστους από το σύνολο η οποία συνδέει μία κορυφή του δέντρου με μία κορυφή εκτός δέντρου.
  - Προσθήκη της ακμής (και της κορυφής) στο δέντρο

# Βασικοί Αλγόριθμοι Γράφων

- **Εντοπισμός Ελάχιστων Δέντρων σε Γράφους**
  - **Σύγκριση Prim vs Kruskal:**




- Ο Kruskal θα επέλεγε την (A, C), την (B, D), την (C, E), θα απέρριπτε την (A, E) επειδή οδηγούσε σε κύκλο με την (A, C) και την (C, E) που έχουν ήδη επιλεγεί, θα επέλεγε την (A, B) και στη συνέχεια θα σταματούσε
  - γιατί είχε ολοκληρώσει τον υπολογισμό ενός πλήρους ελάχιστου δέντρου επικάλυψης

Ο Prim, αν ξεκινούσε από τον κόμβο A, θα τον θεωρούσε κόμβο του δέντρου, και στη συνέχεια θα συμπεριλάμβανε τους C, E, B και D

ο Prim επιστρέφει το ίδιο δέντρο με τον Kruskal

  - οι ακμές επιλέγονται με διαφορετική σειρά.

# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού (Shortest path)
  - επιδιώκεται ο προσδιορισμός του πλέον συντόμου, φθηνότερου ή αξιόπιστου μονοπατιού μεταξύ ενός ή περισσοτέρων ζευγών κόμβων σε ένα δίκτυο.
  - διαφοροποιείται από το πρόβλημα εύρεσης του ελαχίστου δέντρου επικάλυψης
    - το πρώτο προσπαθεί να προσδιορίσει το φθηνότερο μονοπάτι μεταξύ κάποιων κόμβων
    - το δεύτερο αναφέρεται στο φθηνότερο μονοπάτι το οποίο συνδέει όλους τους κόμβους του δικτύου

# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
  - εξετάζεται ένας κατευθυντικός γράφος G=(V,E),
  - Η κάθε ακμή e που ανήκει στο E, έχει ένα βάρος $c_e$.
    - κόστος, χρονική επιβάρυνση, απώλειες, κ.α.
  - Για ένα ζεύγος κόμβων, $u_1$ και $u_k$, **το βάρος ενός μονοπατιού** $P=u_1e_1u_2e_2...u_{k-1}e_{k-1}u_k$, όπου $u_i$ ανήκει στο $V$, και $e_i$ ανήκει στο $E$ είναι το άθροισμα των βαρών των $e_i$ που συμμετέχουν στο μονοπάτι:

$$w(p) = \sum_{i=1}^{k-1} c_{e_i}$$

  - προσδιορισμός του μονοπατιού που ξεκινάει από τον κόμβο $u_1$ και καταλήγει στον $u_k$ έτσι ώστε να ελαχιστοποιηθεί η ποσότητα w(p).

# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
  - τα συντομότερα μονοπάτια εμφωλεύονται (nested)
  - εάν κάποιος κόμβος k αποτελεί τμήμα του συντομότερου μονοπατιού από i στο j, τότε το συντομότερο μονοπάτι (i, j) θα πρέπει να είναι το συντομότερο (i,k) μονοπάτι και το συντομότερο (j,k) μονοπάτι.
  - Τα ελάχιστα μονοπάτια μπορούν να προσδιοριστούν μέσω της αναδρομικής σχέσης:

$$d_{ij} = \min_k(d_{ik} + d_{kj})$$

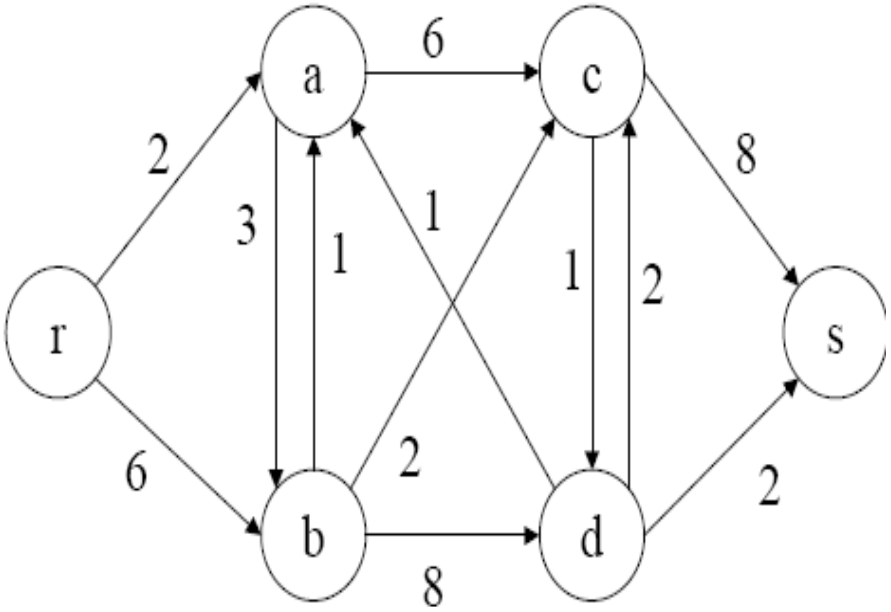
  - όπου $d_{xy}$ είναι το μήκος του συντομότερου μονοπατιού από το x στο y.

# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Dijkstra
  - εφαρμόζεται σε κατευθυνόμενους γράφους με μη-αρνητικά βάρη ακμών.
  - Εντοπίζει τα ελάχιστα μονοπάτια από τον κόμβο αφετηρία r σε όλους τους υπολοίπους κόμβους του γράφου.
  - Βασική ιδέα η αντικατάσταση των προσωρινών ετικετών που αναθέτονται στους κόμβους με μόνιμες.
  - Η μόνιμη ετικέτα ενός κόμβου υποδηλώνει το συνολικό κόστος του ελαχίστου μονοπατιού από τον κόμβο αφετηρία στον τρέχοντα κόμβο.

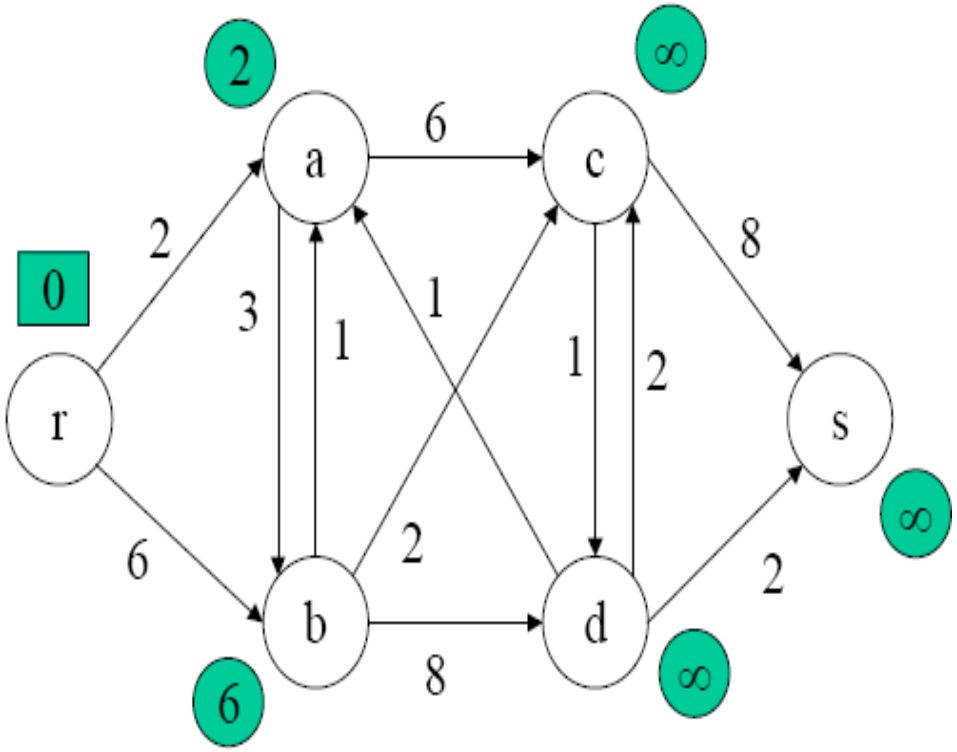
# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Dijkstra



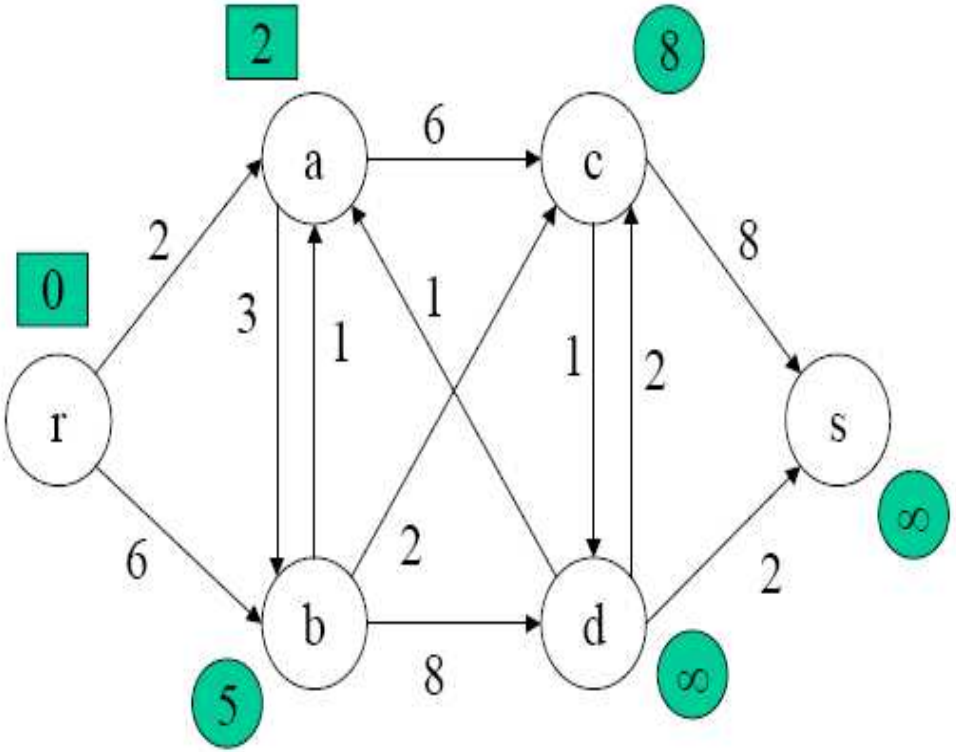
# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Dijkstra



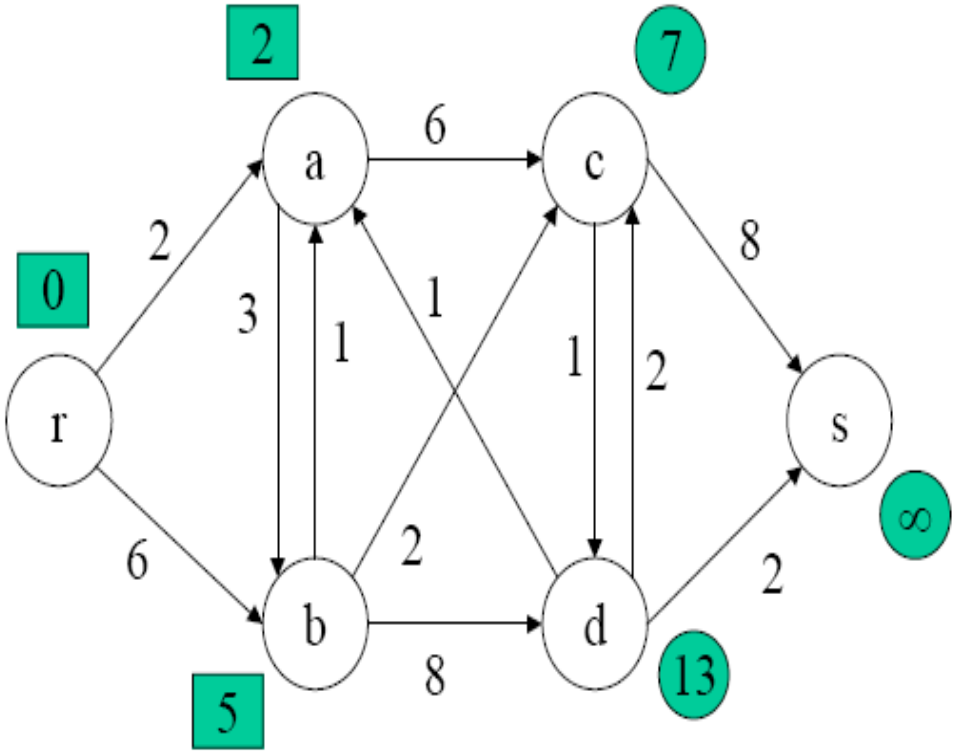
# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Dijkstra



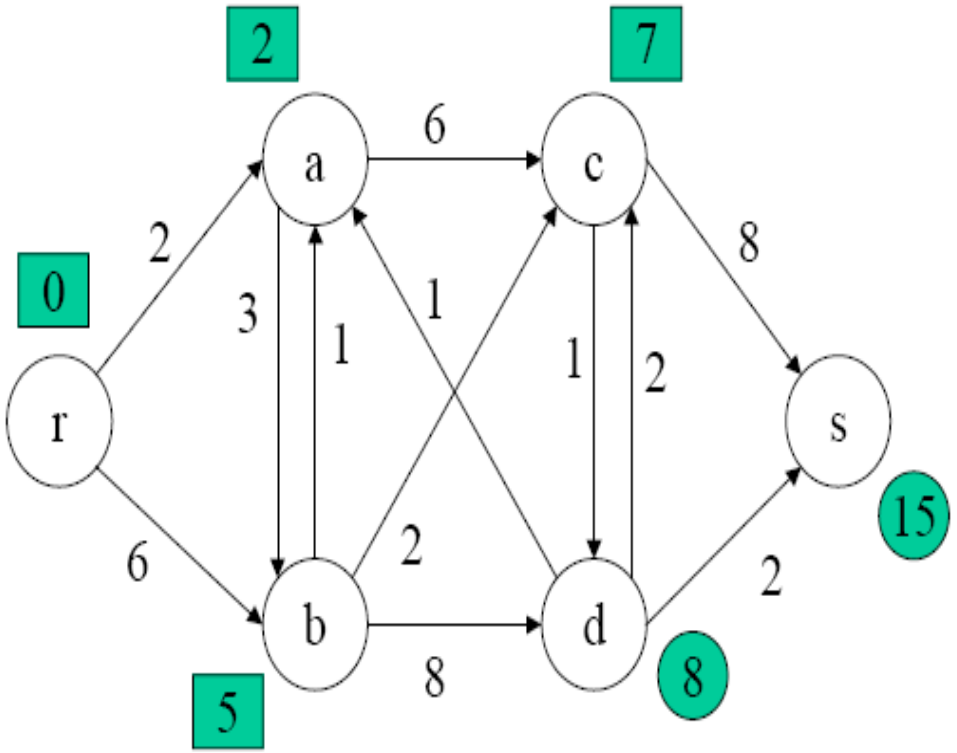
# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Dijkstra



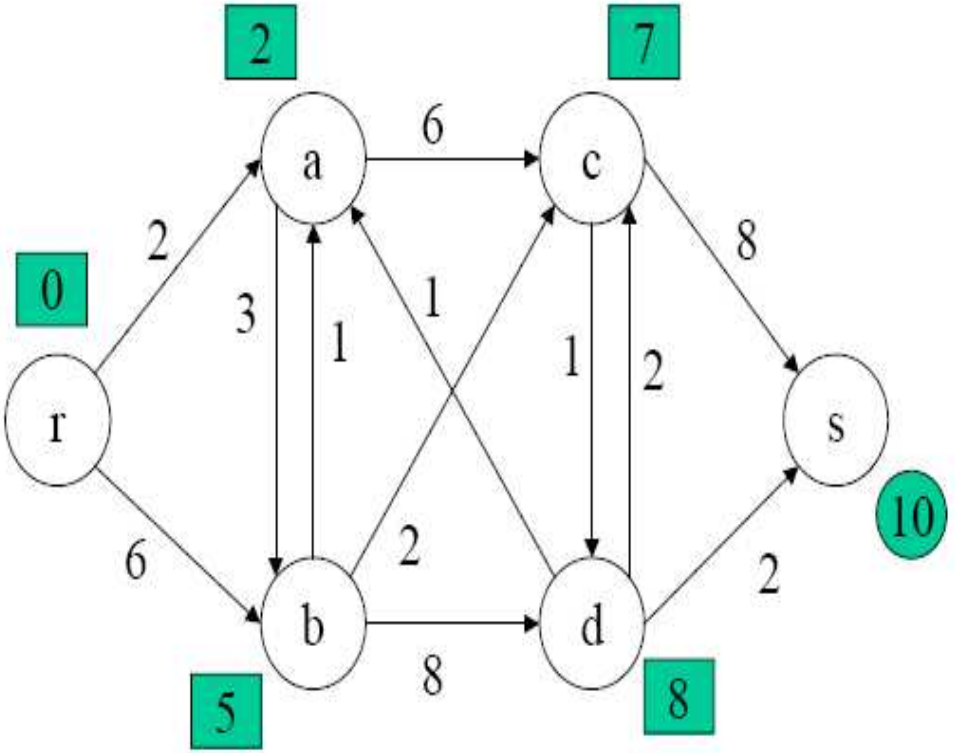
# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Dijkstra



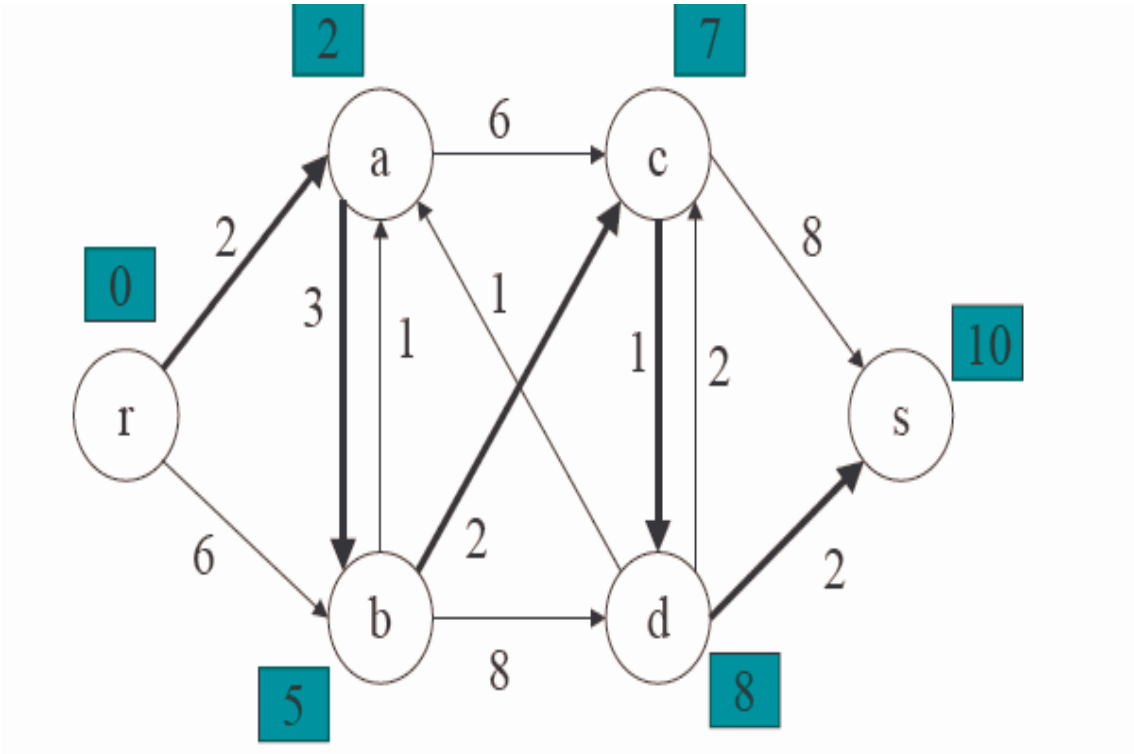
# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Dijkstra



# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Dijkstra

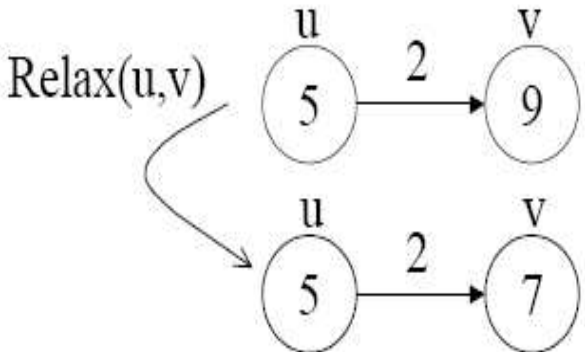


# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Bellman-Ford
  - Υποθέτουμε ότι ο κόμβος 1 είναι ο κόμβος προορισμού (destination) και εξετάζουμε το πρόβλημα εντοπισμού του συντομότερου μονοπατιού από κάθε κόμβο προς τον κόμβο 1.
  - Υποθέτουμε ότι υπάρχει τουλάχιστον ένα μονοπάτι από κάθε κόμβο προς τον προορισμό.
  - Θεωρούμε ότι $d_{ij}=\infty$ thn απόσταση κάθε ακμής (i, j) που δεν ανήκει στον γράφο
  - Το συντομότερο μονοπάτι από ένα κόμβο i προς τον κόμβο 1, με τον περιορισμό ότι το μονοπάτι περιέχει h ή λιγότερες ακμές και διέρχεται από τον κόμβο 1 μόνο μία φορά, καλείται συντομότερο (≤ h) μονοπάτι

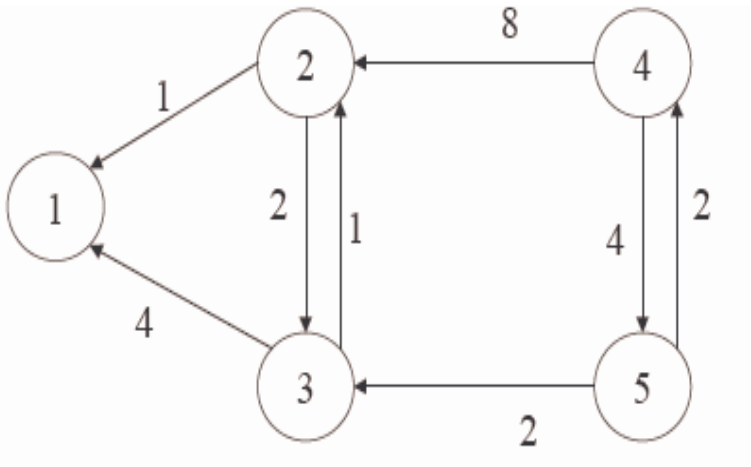
# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Bellman-Ford
  - τεχνική της χαλάρωσης (relaxation).
  - προχωράει στην χαλάρωση μίας ακμής (u, v) όταν μπορεί να βελτιωθεί το συντομότερο μονοπάτι προς τον κόμβο v με την μετακίνηση μέσω του κόμβου u
  - Η εκτίμηση για το συνολικό κόστος του συντομότερου μονοπατιού παρουσιάζεται μέσα στον κάθε κόμβο
  - Στην (α), η τρέχουσα εκτίμηση κόστους συντομότερου μονοπατιού για τον v είναι 9, ενώ η αντίστοιχη εκτίμηση για τον u είναι 5 και το κόστος της ακμής (u,v) είναι 2
  - Κατά συνέπεια, το 9 μπορεί να αντικατασταθεί με την τιμή 5+2=7.



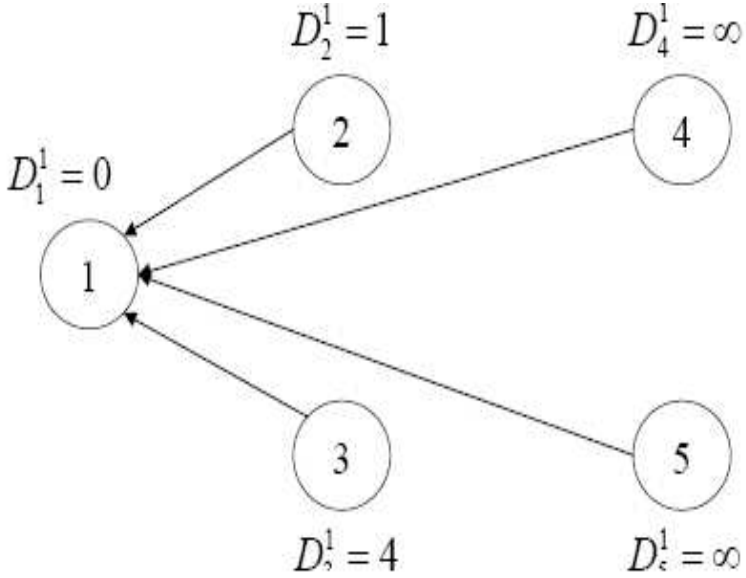
# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Bellman-Ford



# Βασικοί Αλγόριθμοι Γράφων

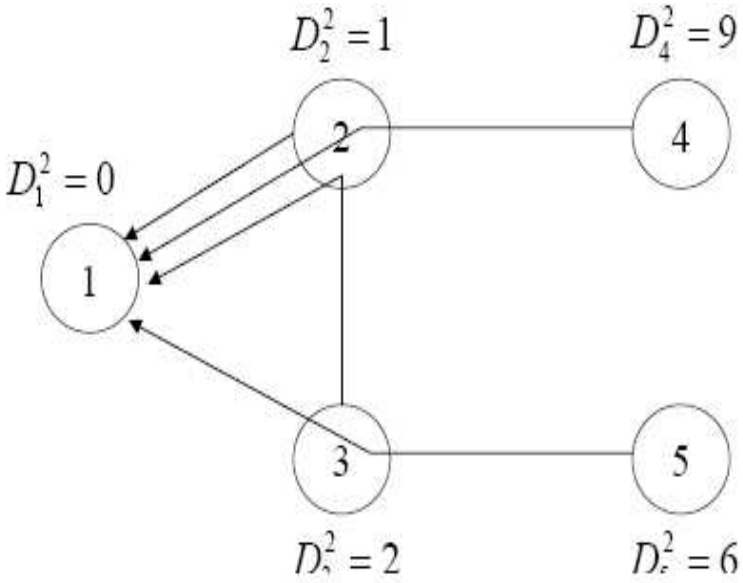
- Πρόβλημα συντομότερου μονοπατιού
- Bellman-Ford




Συντομότερα μονοπάτια χρησιμοποιώντας
1 ακμή ή λιγότερες.

# Βασικοί Αλγόριθμοι Γράφων

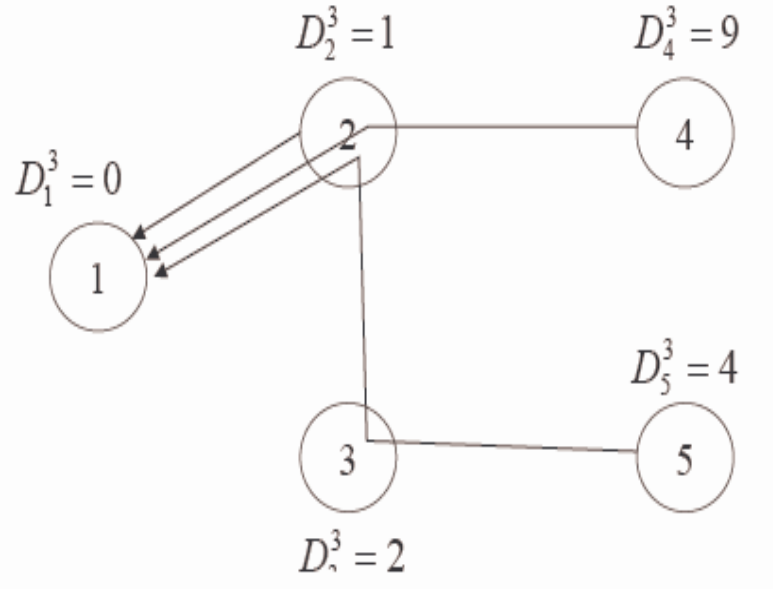
- Πρόβλημα συντομότερου μονοπατιού
- Bellman-Ford




Συντομότερα μονοπάτια χρησιμοποιώντας
2 ακμες ή λιγότερες.

# Βασικοί Αλγόριθμοι Γράφων

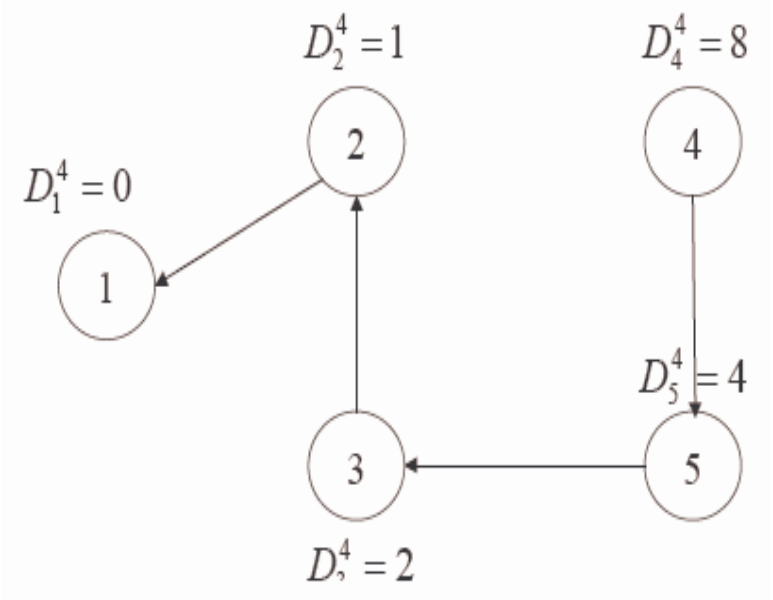
- Πρόβλημα συντομότερου μονοπατιού
- Bellman-Ford




Συντομότερα μονοπάτια χρησιμοποιώντας
3 ακμες ή λιγότερες.

# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Bellman-Ford




Τελικό Δένδρο

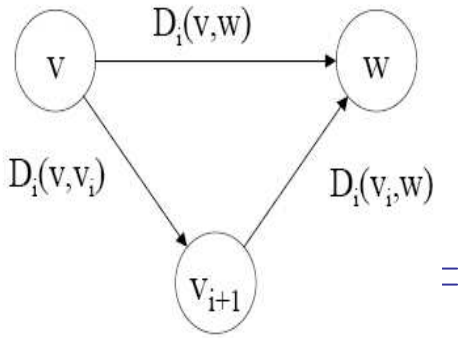
# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Floyd
  - χρησιμοποιεί γραμμικό προγραμματισμό για να επιλύσει το πρόβλημα συντομότερου μονοπατιού **για όλα τα ζεύγη κόμβων** ενός γράφου
  - Η μέθοδος χρησιμοποιεί πίνακα γειτνίασης
  - Για γράφο G=(V,E) έστω C(v,w) το βάρος της ακμής (v,w)
  - Οι κορυφές αριθμούνται από 1 έως |V|. Δηλαδή: V={ $v_1$,$v_2$ ,...,$v_{|N|}$ }
  - Έστω $V_k$ το σύνολο των k πρώτων ακμών στο V
    - Δηλαδή:$V_κ$={ $v_1$,$v_2$ ,...,$v_k$ }
  - Έστω $P_k$(v,w) το συντομότερο μονοπάτι από v προς w που διέρχεται μόνο από τις κορυφές του $V_k$, εάν υπάρχει κάποιο τέτοιο μονοπάτι
  - Δηλαδή το μονοπάτι $P_k$(v,w) είναι της μορφής: { $v_1$,$v_2$ ,...,w }

Ανήκουν στο $V_k$

# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Floyd
  - Έστω $D_k(v,w)$ το μήκος του μονοπατιού $P_k(v,w)$
    - Αν δεν υπάρχει $P_k(v,w)$ το μήκος του θεωρείται άπειρο.
  - Εφόσον, $V_0=\varnothing$, τα μονοπάτια $P_0$ είναι οι κορυφές του γράφου G
    - Τα μήκη $D_0$ των μονοπατιών $P_0$ αντιστοιχούν στα βάρη των ακμών του G.
  - Ο Floyd υπολογίζει διαδοχικά τους πίνακες $D_0$, $D_1$, $D_2$, …. $D_{|V|}$.
  - Οι αποστάσεις στους πίνακες $D_i$ αναπαριστούν μονοπάτια με ενδιάμεσους κόμβους στο σύνολο $V_i$.
  - Εφόσον $V_{i+1}=V_i \cup \{v_{i+1}\}$, οι αποστάσεις που περιέχονται στον $D_{i+1}$ μπορούν να προσδιοριστούν από τον $D_i$ λαμβάνοντας υπόψη μόνο τα μονοπάτια που διέρχονται από τον κόμβο $v_{i+1}$.

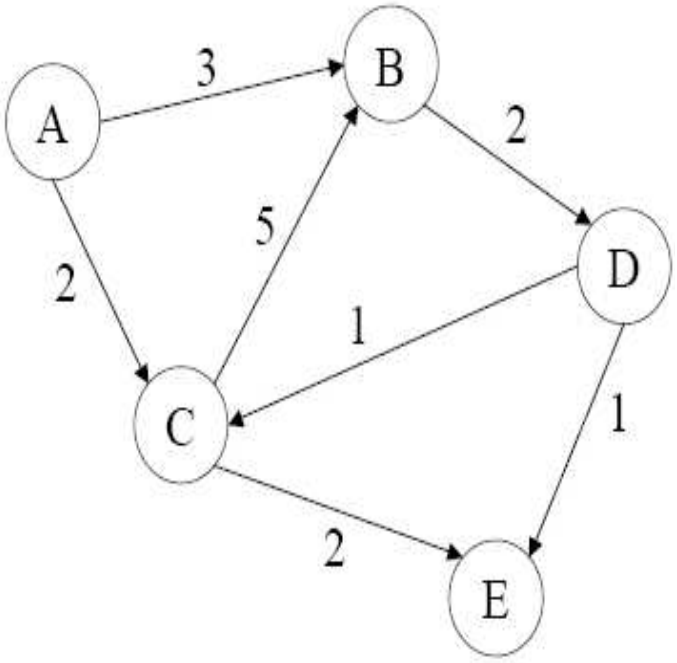


# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Floyd
  - Για κάθε ζεύγος κόμβων (v,w) συγκρίνονται οι αποστάσεις Di(v,w) (που αναπαριστά τα συντομότερο μονοπάτι από τον κόμβο v στο κόμβο w χωρίς να μεσολαβεί ο $v_{i+1}$) με το άθροισμα $Di(v,v_{i+1})+Di(v_{i+1},w)$ (που αναπαριστά το συντομότερο μονοπάτι από το v στο w που διέρχεται από το $v_{i+1}$. Έτσι, το $D_{i+1}$ υπολογίζεται ως εξής: $$D_{i+1}(v,w)=\min\{D_i(v,v_{i+1})+D_i(v_{i+1},w),D_i(v,w)\}$$
  - πριν την ενεργοποίηση αλγορίθμου θα πρέπει να διαμορφωθεί ένας πίνακας n×n ( n πλήθος κόμβων γράφου).
  - κάθε γραμμή αναπαριστά ένα κόμβο αφετηρίας στο γράφο ενώ στήλη αναπαριστά τα σημεία τερματισμού.
  - Εάν υπάρχει ακμή μεταξύ σημείου αφετηρίας και σημείου τερματισμού, το σχετικό κόστος τοποθετείται στην θέση (i, j)

# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Floyd




| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 0 | 3 | 2 | ∞ | ∞ |
| B | ∞ | 0 | ∞ | 2 | ∞ |
| C | ∞ | 5 | 0 | ∞ | 2 |
| D | ∞ | ∞ | 1 | 0 | 1 |
| E | ∞ | ∞ | ∞ | ∞ | 0 |

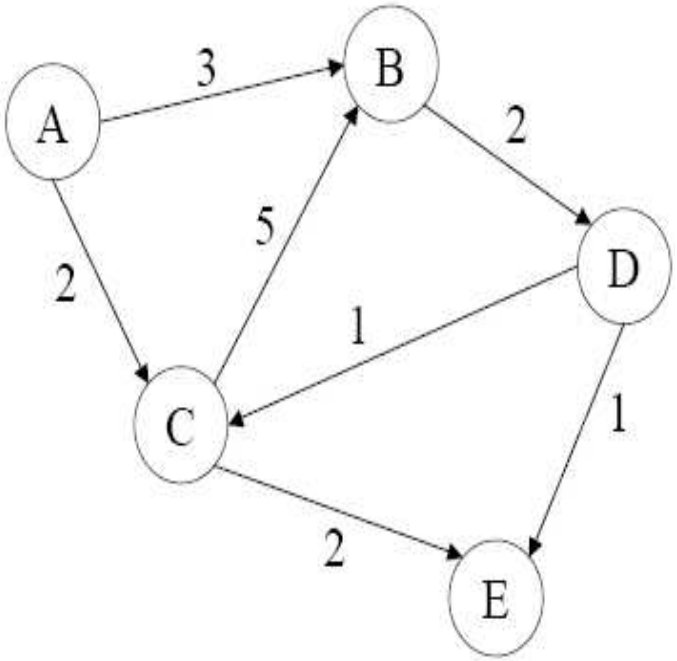
**αρχικός πίνακας αποστάσεων μεταξύ κόμβων. Οι κόμβοι στήλης αναπαριστούν προορισμούς (to) οι κόμβοι γραμμής αναπαριστούν αφετηρίες (from).**

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | A | A | A | A | A |
| B | B | B | B | B | B |
| C | C | C | C | C | C |
| D | D | D | D | D | D |
| E | E | E | E | E | E |

**αρχική κατάσταση πίνακα προηγούμενου κόμβου (predecessor)**

# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Floyd




| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 0 | 3 | 2 | 5 | ∞ |
| B | ∞ | 0 | ∞ | 2 | ∞ |
| C | ∞ | 5 | 0 | 7 | 2 |
| D | ∞ | ∞ | 1 | 0 | 1 |
| E | ∞ | ∞ | ∞ | ∞ | 0 |

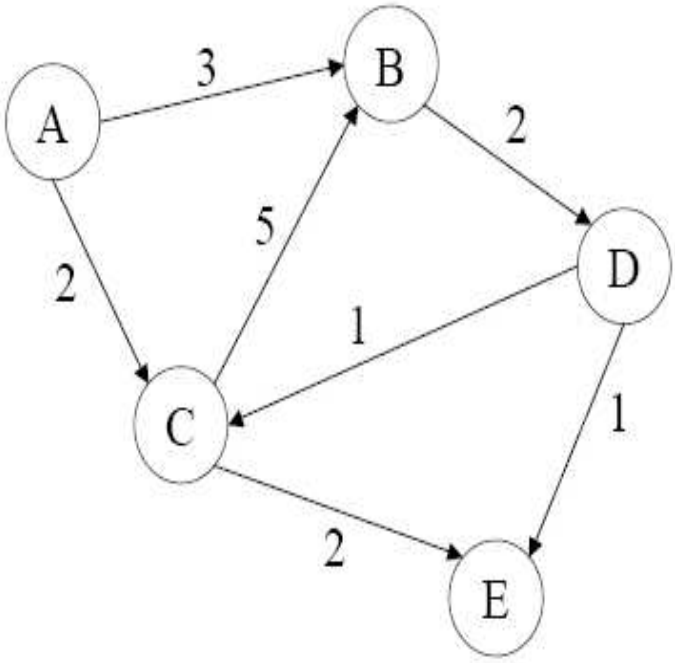
**Αν θεωρήσουμε τον κόμβο Β ως ενδιάμεσο κόμβο, οι τιμές στις θέσεις (A, D) και (C, D) του πίνακα αλλάζουν**

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | A | A | A | B | A |
| B | B | B | B | B | B |
| C | C | C | C | B | C |
| D | D | D | D | D | D |
| E | E | E | E | E | E |

**νέα κατάσταση πίνακα προηγούμενου κόμβου (predecessor)**

# Βασικοί Αλγόριθμοι Γράφων

- Πρόβλημα συντομότερου μονοπατιού
- Floyd




| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 0 | 3 | 2 | 5 | 4 |
| B | ∞ | 0 | 3 | 2 | 3 |
| C | ∞ | 5 | 0 | 7 | 2 |
| D | ∞ | 6 | 1 | 0 | 1 |
| E | ∞ | ∞ | ∞ | ∞ | 0 |

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | A | A | A | B | C |
| B | B | B | D | B | D |
| C | C | C | C | B | C |
| D | D | C | D | D | D |
| E | E | E | E | E | E |

**αν θεωρηθούν ως ενδιάμεσοι οι κόμβοι C, D και E προκύπτουν οι δίπλα πίνακες:**

# Βασικοί Αλγόριθμοι Γράφων

- Ροές Δικτύων
  - Δοθέντος δικτύου και απαίτησης μεταφοράς ποσότητας αγαθού r από αφετηρία s σε προορισμό d να προσδιοριστεί εφικτός συνδυασμός ροών που δεν υπερβαίνουν τις χωρητικότητες των ζεύξεων
  - Οι ζεύξεις στο δίκτυο είναι οι lij και οι σχετικές χωρητικότητες cij.
  - Το πρόβλημα είναι ο προσδιορισμός ενός η περισσότερων μονοπατιών από το s στο d μέσω των οποίων θα μεταφερθεί η συγκεκριμένη ποσότητα αγαθού r.
  - Το άθροισμα των ροών ισούται με το μέγεθος r
  - Το άθροισμα των ροών σε κάθε ζεύξη δεν θα πρέπει να ξεπερνάει την χωρητικότητα της ζεύξης (< cij για κάθε i,j)

# Βασικοί Αλγόριθμοι Γράφων

- Ροές Δικτύων
- Αλγόριθμος Ford-Fulkerson
  - προσδιορίζει μονοπάτια s-d (από αφετηρία σε προορισμό) και στέλνει μέσω αυτών όση περισσότερη ροή χωρίς να υπερβαίνει τους περιορισμούς της χωρητικότητας
- Θεώρημα Μέγιστης Ροής - Ελάχιστης Τομής (max flow - minimum cut) των Ford- Fulkerson:
  - **Η μέγιστη ροή S-D (από την αφετηρία προς τον προορισμό) είναι ακριβώς ίση με τη χωρητικότητα της ελάχιστης τομής S-D.**

# Βασικοί Αλγόριθμοι Γράφων

- Ροές Δικτύων
- Αλγόριθμος Ford-Fulkerson
  - Ο αλγόριθμος έχει δύο τμήματα: Ρουτίνα Α και Β
  - Η **Α** αναθέτει ετικέτες και ερευνά για μονοπάτι επαύξησης ροής από αφετηρία s σε προορισμό d για το οποίο ισχύει
    - f < c για όλες τις ακμές προς τον προορισμό (forward arcs) και
    - f > 0 σε όλες τις ακμές αντίθετης κατεύθυνσης (backward arcs)
  - Εάν η ρουτίνα Α εντοπίσει μονοπάτι επαύξησης ροής, η Ρουτίνα Β μεταβάλει τη ροή αντίστοιχα.
  - Διαφορετικά η τρέχουσα ροή είναι βέλτιστη
    - Αρχικά επιλέγεται εφικτή ροή (π.χ., f=0).
  - Ένας κόμβος βρίσκεται σε μία από τρεις καταστάσεις:
    - **χωρίς ετικέτα**, **ελεγμένος με ετικέτα**, **μη- ελεγμένος με ετικέτα**.
    - Αρχικά όλοι οι κόμβοι είναι χωρίς ετικέτα

# Βασικοί Αλγόριθμοι Γράφων

- Ροές Δικτύων
- Αλγόριθμος Ford-Fulkerson - Ρουτίνα A
  - Αρχικά η αφετηρία παίρνει την ετικέτα (-,ε(s)=∞).
  - Γενικό βήμα: Επιλογή κόμβου x μη-ελεγμένο με ετικέτα.
    - Η ετικέτα του είναι η (z±,ε(x)).
  - Σε όλους τους διάδοχους κόμβους (χωρίς ετικέτα), y, και ισχύει η συνθήκη f(x, y) < c(x, y, ανατίθενται οι ετικέτες (x+, ε(y)) όπου

    $$\varepsilon(y) = \min(\varepsilon(x), c(x, y) - f(x, y))$$

    - Οι κόμβοι αυτοί θεωρούνται με ετικέτα και μη ελεγμένοι.
  - Για όλους τους πρόδρομους κόμβους (χωρίς ετικέτα), y, και ισχύει f(y, x) > 0, ανατίθενται οι ετικέτες (x-, ε(y)), όπου

    $$\varepsilon(y) = \min(\varepsilon(x), f(y, x))$$

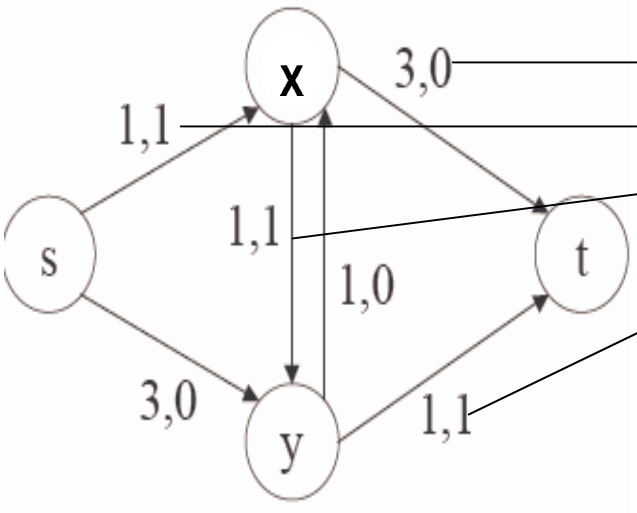
  - Οι κόμβοι y θεωρούνται πλέον με ετικέτα και μη ελεγμένοι.
  - Πλέον ο κόμβος x θεωρείται με ετικέτα και ελεγμένος.
  - Το γενικό βήμα επαναλαμβάνεται μέχρι να ανατεθεί ετικέτα στον προορισμό και να είναι μη-ελεγμένος ή να μην είναι δυνατόν να εκχωρηθούν νέες ετικέτες

# Βασικοί Αλγόριθμοι Γράφων

- Ροές Δικτύων
- Αλγόριθμος Ford-Fulkerson - Ρουτίνα Β (αλλαγής ροής)
  - Ο προορισμός έχει λάβει την ετικέτα (y±,ε(t)).
  - Εάν το πρώτο σκέλος της ετικέτας είναι y+, η τιμή f(y,t) αντικαθίσταται με την τιμή f(y,t)+ε(t) διαφορετικά η τιμή f(t,y) αντικαθίσταται με την τιμή f(t,y)-ε(t).
  - Στη συνέχεια, ο κόμβος x θα αντιμετωπιστεί με τον ίδιο τρόπο:
    - εάν η ετικέτα του είναι (x+, ε(y)) η τιμή f(x, y) αντικαθίσταται με την f(x, y) + ε(t).
    - Εάν η ετικέτα είναι (x-, ε(y)) η τιμή f(y, x) αντικαθίσταται με την f(y, x)-ε(t).
  - Και στις δύο περιπτώσεις, θα συνεχιστεί η εκτέλεση του αλγορίθμου στον κόμβο x και να συνεχιστεί μέχρι να φτάσει στην αφετηρία. Τότε θα πρέπει να απορριφθούν όλες οι ετικέτες και να ενεργοποιηθεί η ρουτίνα Α.

# Βασικοί Αλγόριθμοι Γράφων

- Ροές Δικτύων
- Αλγόριθμος Ford-Fulkerson – Παράδειγμα

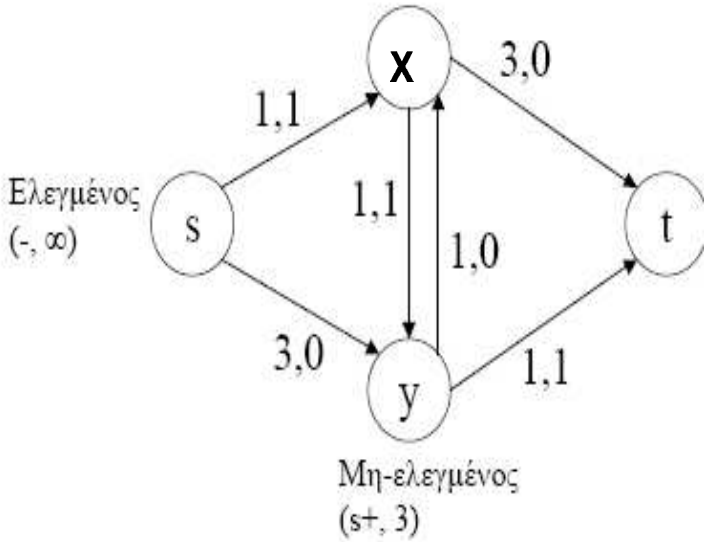



χωρητικότητα (c(x,y)) καθώς και ροή (f(x,y)).

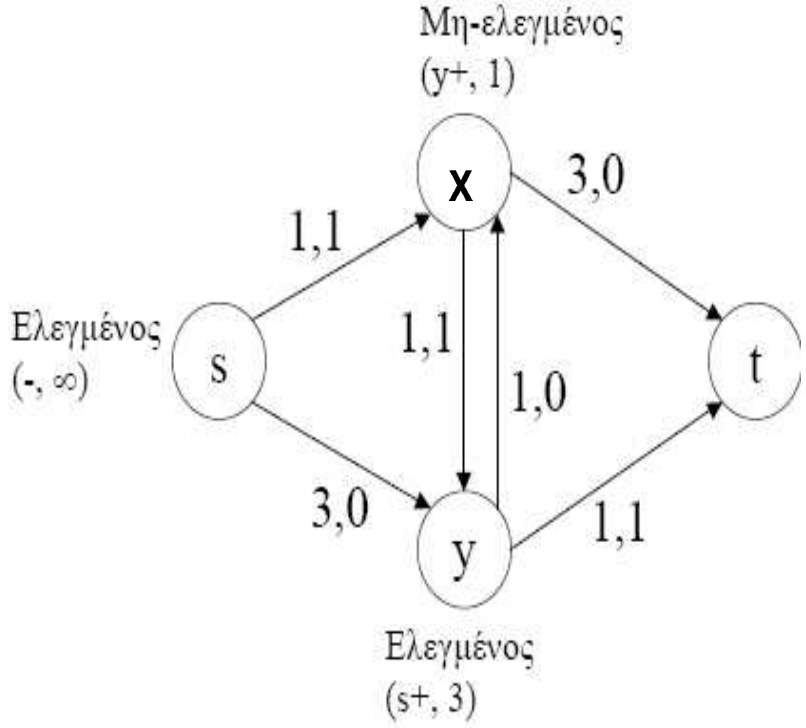
Η αρχική ροή στέλνει μία μονάδα ροής από το μονοπάτι (s, x, y, t).

# Βασικοί Αλγόριθμοι Γράφων

- Ροές Δικτύων
- Αλγόριθμος Ford-Fulkerson - Παράδειγμα



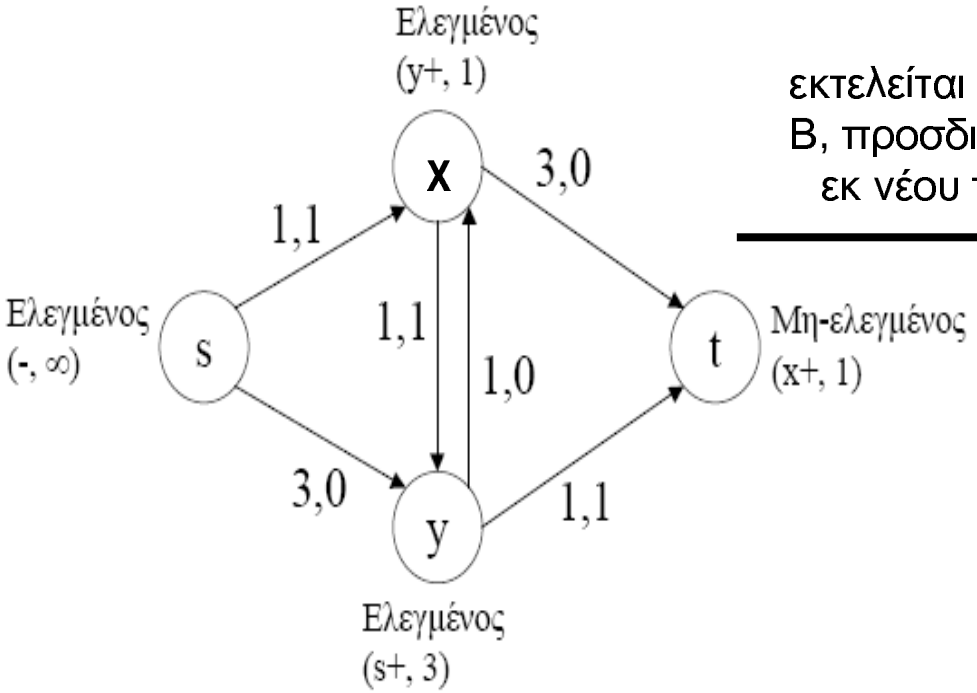

Η πρώτη εκτέλεση της Ρουτίνας Α προσδιορίζει ένα μονοπάτι επαύξησης ροής με τις ετικέτες που φαίνονται στο σχήμα




Μετά το 2ο βήμα της Α

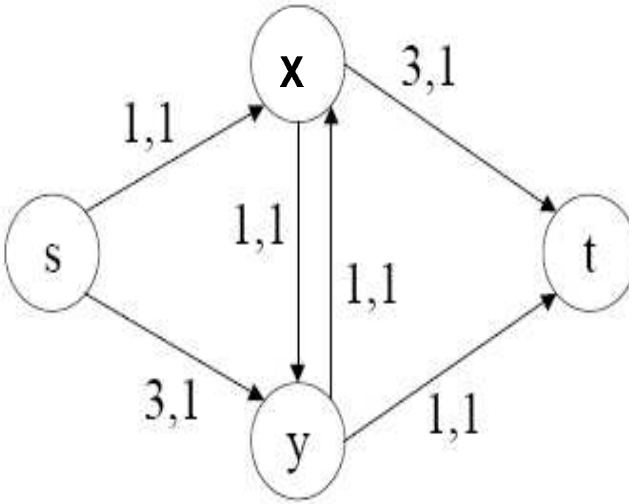
# Βασικοί Αλγόριθμοι Γράφων

- Ροές Δικτύων
- Αλγόριθμος Ford-Fulkerson - Παράδειγμα




Μετά το 2ο βήμα της A

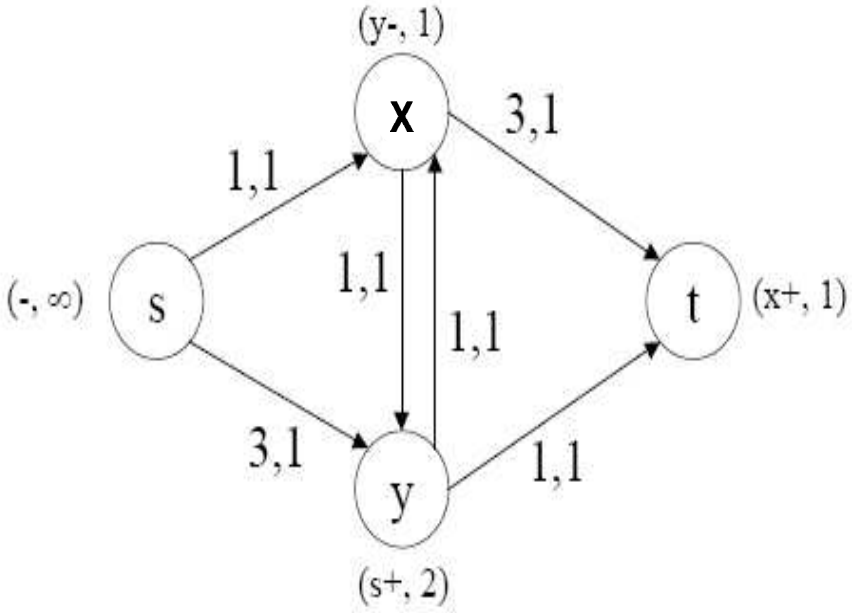
εκτελείται η Ρουτίνα
Β, προσδιορίζοντας
εκ νέου τις ροές.




Ρουτίνα Β - Προσδιορισμός ροών

# Βασικοί Αλγόριθμοι Γράφων

- Ροές Δικτύων
- Αλγόριθμος Ford-Fulkerson - Παράδειγμα




Από το δίπλα σχήμα φαίνεται ότι η ροή μπορεί να αυξηθεί κατά μία μονάδα (=ε(t)) κατά μήκος του επαυξητικού μονοπατιού (s, (y, x), t).

Εκτελείται και πάλι η Ρουτίνα Α δίνοντας τις ετικέτες που φαίνονται στο Σχήμα

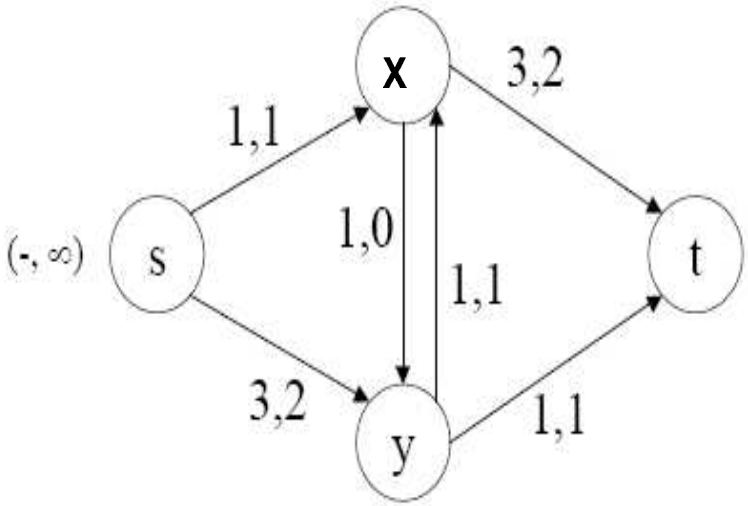
# Βασικοί Αλγόριθμοι Γράφων

- Ροές Δικτύων
- Αλγόριθμος Ford-Fulkerson - Παράδειγμα

Κατά την εφαρμογή του αλγορίθμου αντίστροφα, από τον προορισμό, το μονοπάτι διασχίζει την ακμή (x, t), οπότε η ροή του αυξάνεται κατά 1 (η ποσότητα f(x,t) γίνεται 2).

Η επικεφαλίδα στο κόμβο x υποδεικνύει τη διάσχιση της ακμής (x,y) με κατεύθυνση προς τα πίσω, οπότε η ροή της μειώνεται κατά 1 (η ποσότητα f(x,y) γίνεται 0). Στην συνέχεια, η ετικέτα του κόμβου y, μας υποδεικνύει ότι το μονοπάτι επαύξησης ροής διασχίζει την ακμή (s,y) με κατεύθυνση προς τα εμπρός, οπότε η ροή του αυξάνεται κατά 1 (η ποσότητα f(s,y) γίνεται 2).

Εφόσον, ο αλγόριθμος έφτασε στην αφετηρία, τερματίζεται η εκτέλεση της Ρουτίνας Β και εκτελείται πάλι η Ρουτίνα Α με τις ακόλουθες τιμές ροής και αρχικές ετικέτες.

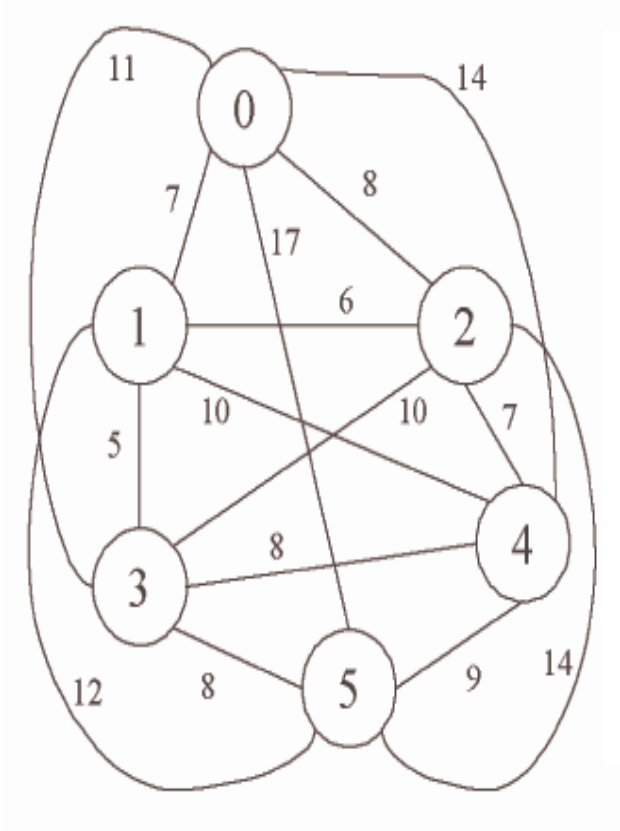


# Βασικοί Αλγόριθμοι Γράφων

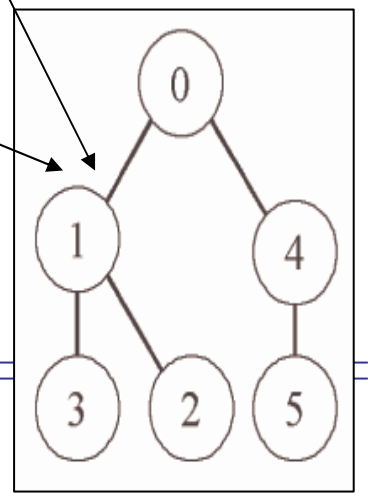
- **Σχεδιασμός Γραμμών Πολλαπλών Σημείων**
- Ελάχιστα δέντρα επικάλυψης υπό περιορισμούς
  - Πρόβλημα ΕΔΕ με περιορισμό στο μέγεθος των υπο-δέντρων
    - Constrained Minimum Spanning Tree – CMST
  - Μπορεί να χρησιμοποιηθεί τροποποιημένος Kruskal
  - Είναι άπληστος
  - Ανήκει στην οικογένεια αλγορίθμων καλύτερης ακμής
  - Επιλέγει την καλύτερη εφικτή ακμή και συνδέουν μέσω αυτής τα συστατικά του γράφου

# Βασικοί Αλγόριθμοι Γράφων

- **Σχεδιασμός Γραμμών Πολλαπλών Σημείων**
- Ελάχιστα δέντρα επικάλυψης υπό περιορισμούς – Kruskal




- Βάρος κάθε κόμβου = 1, μέγιστο βάθος W=3
- Επιλέγονται οι ακμές (1, 3), (1, 2) και (0, 1), με αυτή τη σειρά (αποτελούν τμήμα του MST)
- Οι ακμές (2, 4) και (3, 5) (θα ολοκλήρωναν το MST) απορρίπτονται λόγω W=3
- Ο γράφος ολοκληρώνεται με τις (4, 5) και (4, 0).
- Συνολικό κόστος δικτύου 41
- Υπάρχουν δέντρα χαμηλότερου κόστους
  - Π.χ. [(0, 1), (1, 3), (0, 2), (2, 4), (4, 5)] με κόστος 36.



# Δρομολόγηση

- **Στόχος**
  - Λήψη απόφασης για τον τρόπο μεταφοράς απαιτήσεων κυκλοφορίας στο δίκτυο
  - Υπάρχουν περισσότερα από ένα μονοπάτια για την εξυπηρέτηση της ανάγκης
  - Υπάρχει
    - τοπολογία
    - Χωρητικότητες κόμβων
    - Κόστη γραμμών μετάδοσης
      - Καθυστέρηση, απώλεια, κοκ
  - Επιλογή με βάση κριτήρια απόδοσης

# Δρομολόγηση

- **Διαδικασίες Δρομολόγησης**
- Πλημμύρα – flooding
  - Κάθε κόμβος στέλνει αντίγραφο πακέτου που έλαβε σε όλους τους γείτονες
  - Δεν είναι ανάγκη ο κόμβος να γνωρίζει τοπολογία δικτύου
  - Περιορισμός αέναης κυκλοφορίας του ίδιου πακέτου στο δίκτυο
    - Time-to-leave
    - Καταγραφή πακέτου που έχει ληφθεί
  - Πλεονέκτηματα
    - Όχι ανάγκη αλλαγής πληροφοριών κατάστασης
    - Απλότητα και ταχύτητα
  - Μειονέκτημα
    - Διπλότυπα πακέτα – σπατάλη πόρων

# Δρομολόγηση

- **Διαδικασίες Δρομολόγησης**
- Ρητή δρομολόγηση
  - Εκ των προτέρων ορισμός διαδρομής για κάθε ζεύγος προορισμού – αφετηρίας
  - Διαφορετικές διαδρομές για διαφορετικά είδη κυκλοφορίας
  - Δεν είναι ανάγκη ο κόμβος να γνωρίζει τοπολογία δικτύου
  - Πλεονέκτηματα
    - Όχι ανάγκη αλλαγής πληροφοριών κατάστασης
    - Απλότητα και ταχύτητα
  - Μειονέκτημα
    - Διαχείριση σε μεγάλης κλίμακας δίκτυα

# Δρομολόγηση

- **Διαδικασίες Δρομολόγησης**
- Στατική δρομολόγηση
  - Κάθε ζεύξη και κόμβος χαρακτηρίζονται από μήκος ή κόστος
  - Συνήθως αναλλοίωτα στο χρόνο
  - Για ανάθεση ετικέτας κόστους από τον κόμβο j στον i υπολογίζεται η $D_j = D_i + L_i + L_{ij}$
    - Di: μήκος μονοπατιού μέχρι κόμβο i
    - Li: κόστος κόμβου i
    - Lij: μήκος ζεύξης από j σε i
  - Αν όλες οι ζεύξεις έχουν τιμή κόστους 1, τότε δρομολόγηση ελάχιστων βημάτων
  - Αν οι ζεύξεις έχουν τιμή κόστους, τότε δρομολόγηση ελάχιστου κόστους

# Δρομολόγηση

- **Διαδικασίες Δρομολόγησης**
- Προσαρμοστική δρομολόγηση
  - Συμφόρηση και βλάβες
  - Το κόστος (μήκος) ζεύξεων και κόμβων αυξάνει με τον κορεσμό
  - Διάχυση πληροφορίας συμφόρησης πολλές φορές αδύνατη
  - Η προσαρμοστική δρομολόγηση κατευθύνει κίνηση εκτός των κορεσμένων μονοπατιών.
  - Αν η εικόνα που συντηρούν οι κόμβοι δικτύου είναι παρωχημένη τότε μη αποτελεσματική δρομολόγηση
    - ταλάντωση συμφόρησης
    - Διαμόρφωση βρόγχων
    - ➔ Κατάρρευση δικτύου

# Δρομολόγηση

- **Διαδικασίες Δρομολόγησης**
- Κατανεμημένη δρομολόγηση
  - Κάθε κόμβος αποφασίζει για τη ζεύξη στην οποία θα προωθήσει το μήνυμα
    - next hop
  - Η διαδικασία επαναλαμβάνεται από όλους τους κόμβους
    - μέχρι το μήνυμα να φθάσει στον προορισμό
  - Απαιτεί συντονισμό μεταξύ κόμβων, εισάγει καθυστερήσεις, και επιβαρύνσεις στους κόμβους

# Δρομολόγηση

- **Διαδικασίες Δρομολόγησης**
- Αλγόριθμος απόκλισης ροής
  - Προσδιορισμός διαδρομών που ικανοποιούν την από-άκρο-σε-άκρο ελάχιστη καθυστέρηση
    - Δεδομένης της τοπολογίας του δικτύου και της ανάγκης μεταφοράς κίνησης
  - Κίνηση εμφανίζεται ανεξάρτητα, μήκη εκθετικά κατανεμημένα → M/M/1
  - Θεωρούμε καθυστέρηση μόνο λόγω αναμονής (και όχι επεξεργασίας)
  - Έστω rij η απαίτηση για μεταφορά κίνησης και k οι διαθέσιμες ζεύξεις με χωρητικότητα $c_k$. Θέλουμε ελαχιστοποίηση T

$$\bar{T} = \frac{1}{\gamma}\sum_{i,j} d_{ij} r_{ij}$$ **όπου** $$\gamma = \sum_{i,j} r_{ij}$$ **και $d_{ij}$ η καθυστέρηση σε όλες τις ζεύξεις**

# Δρομολόγηση

- **Διαδικασίες Δρομολόγησης**
- Αλγόριθμος απόκλισης ροής
  - Η συνολική καθυστέρηση T είναι $\overline{T}=T*\gamma$
  - Ισχύει για το d $d_{ij} = \sum_{p} \sum_{k \in p} a_{ijp} t_k$

όπου p είναι ένα μονοπάτι, $t_k$ η καθυστέρηση στη ζεύξη k και $\alpha_{ijp}$ το ποσοστό της κινησης $r_{ij}$ στο p.

  - Η συνολική καθυστέρηση είναι $T = \sum_{i,j} \sum_{p} \sum_{k \in p} a_{ijp} t_k r_{ij} = \sum_{k} f_k t_k$ όπου $f_k$ η ροή στην ζεύξη k

# Δρομολόγηση

- **Διαδικασίες Δρομολόγησης**
- Αλγόριθμος απόκλισης ροής
  - Πολλές προτάσεις για δρομολόγηση με ελαχιστοποίηση καθυστέρησης
  - Όλες παράγουν αρχικά εφικτή λύση
  - Στη συνέχεια μετακίνηση ροής από κορεσμένα μονοπάτια σε άλλα, λιγότερο κορεσμένα
  - Ο αλγόριθμος απόκλισης ροής κάνει ακριβώς αυτό
  - Ιδέα: Αν μικρό ποσοστό κίνησης μετακινηθεί από μια διαδρομή με αυξημένη καθυστέρηση σε μια άλλη διαδρομή με μικρότερη καθυστέρηση τότε η συνολική καθυστέρηση μειώνεται

# Δρομολόγηση

- **Διαδικασίες Δρομολόγησης**
- Αλγόριθμος απόκλισης ροής
  - Στο M/M/1 η καθυστέρηση σε ζεύξη k με χρόνο εξυπηρέτησης $T_{sk}$ και χρήση $U_k$ είναι

$$t_k = \frac{T_{sk}}{1 - U_k}$$

  - Μάλιστα $T_{sk}$ είναι το μέσο μήκος μηνύματος L, διαιρεμένο με τη χωρητικότητα $c_k$ της ζεύξης, και η χρήση $U_k$ είναι $f_k$, που εξυπηρετείτε από την ζεύξη k διαιρεμένη με $c_k$. Επομένως:

$$t_k = \frac{L/c_k}{1 - f_k/c_k} = \frac{L}{c_k - f_k}$$