Γραφήματα, BFS, DFS, εφαρμογές

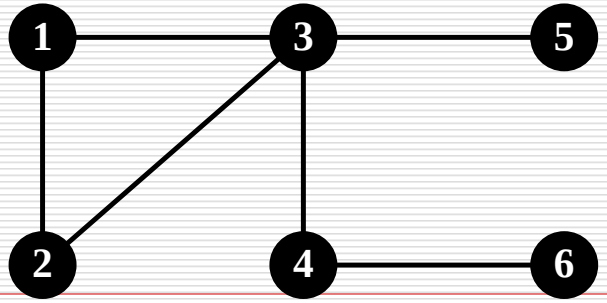
Δημήτρης Φωτάκης, Αριστείδης Παγουρτζής,
Δώρα Σούλιου, Παναγιώτης Γροντάς

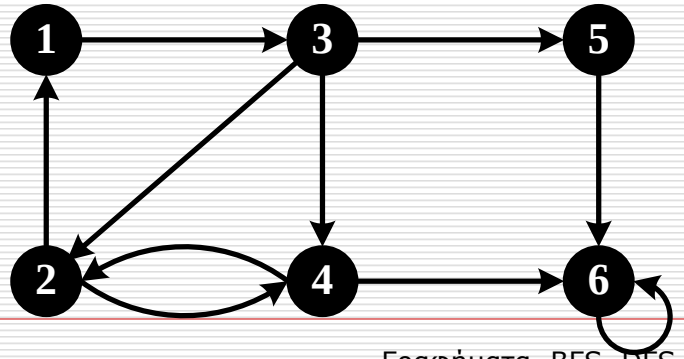
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών

Εθνικό Μετσόβιο Πολυτεχνείο

Γραφήματα

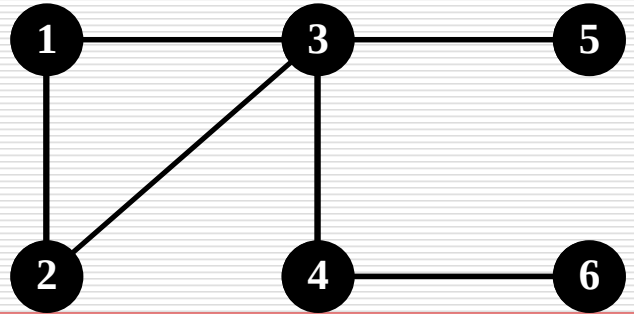
- Αναπαράσταση σχέσεων μεταξύ οντοτήτων
- Μοντελοποίηση πολλών σημαντικών προβλημάτων
  (π.χ. Web, Κοινωνικά δίκτυα, δρομολόγηση, προτεραιότητες)
- Γράφημα $G(V, E)$: V κορυφές - E ακμές (ζεύγη σχετιζόμενων κορυφών)
  - Τάξη $|V| = n$ και μέγεθος $|E| = m$.
  - Κατευθυνόμενα και μη-κατευθυνόμενα
  - Πολυγράφημα: Παράλληλες ακμές
  - Βάρη (μήκη) στις ακμές $G(V, E, w)$, $w : E \mapsto \mathbb{R}$

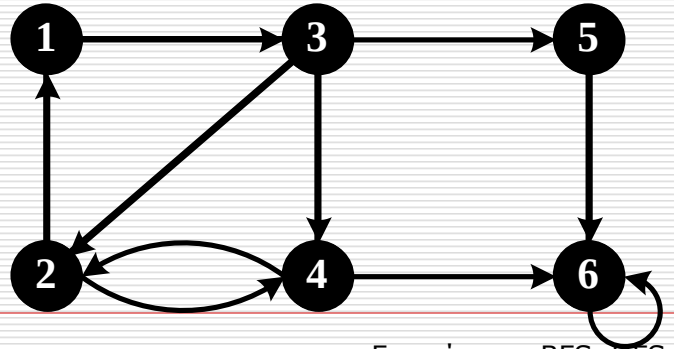




Γραφήματα

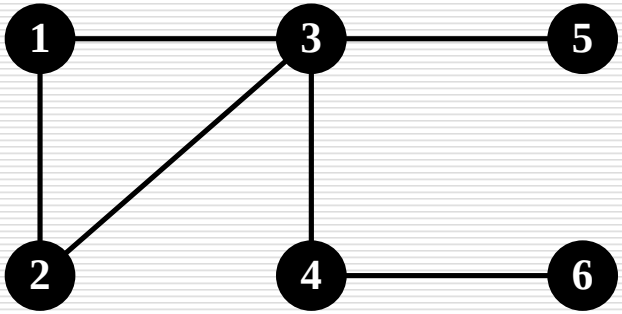
- Βαθμός κορυφής $\deg(v)$: #ακμών εφαπτόμενων στη u.
  - Κατευθυνόμενα: εισερχόμενος και εξερχόμενος βαθμός.
  - Μη-κατευθυνόμενο G(V, E): $\sum_{v \in V} \deg(v) = 2|E|$
- Διαδρομή, μονοκονδυλιά, μονοπάτι (απλό).
- Κλειστή διαδρομή, κύκλωμα, κύκλος (απλός).
- Απόσταση $d(u, v)$ (χωρίς και με βάρη).
- Συνεκτικό: μονοπάτι μεταξύ κάθε ζεύγους κορυφών.
- Δέντρο: ακυκλικό συνεκτικό γράφημα. Δάσος.

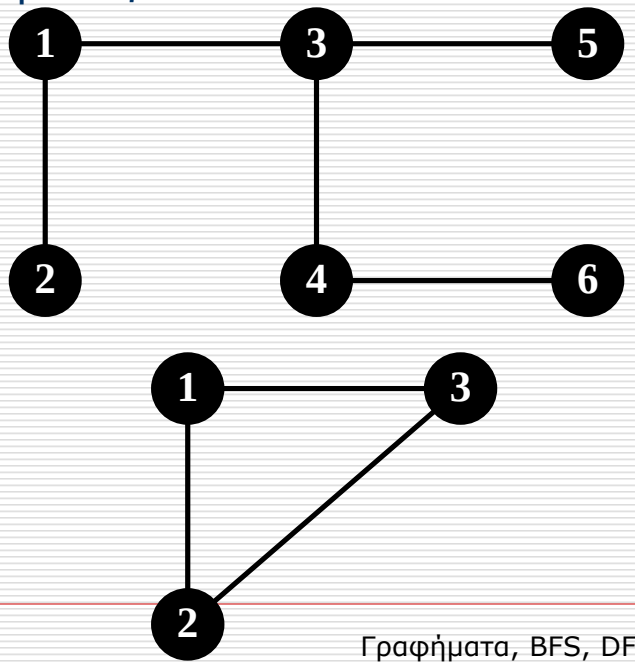




Υπο-Γραφήματα

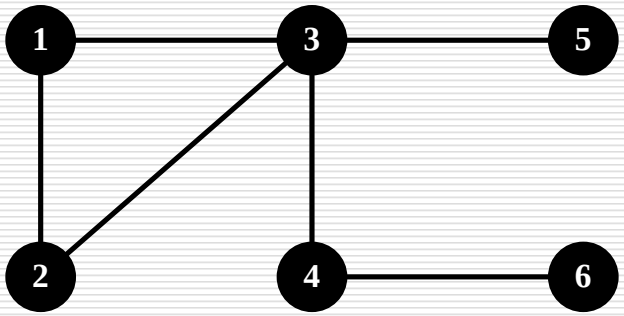
- Υπογράφημα $G'(V', E')$ του $G(V, E)$ όταν $V' \subseteq V$ και $E' \subseteq E$.
  - Επικαλύπτον (spanning) όταν $V' = V$,
    δηλ. έχει όλες τις κορυφές του αρχικού γραφήματος.
  - Επαγόμενο (induced) όταν $E' = \{(u, v) \in E : u, v \in V'\}$
    δηλ. έχει όλες τις ακμές του αρχικού μεταξύ
    των επιλεγμένων κορυφών.





Αναπαράσταση Γραφημάτων

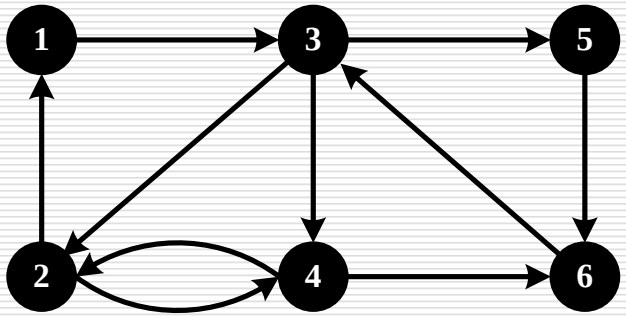
- ... με **πίνακα γειτνίασης:** $A[i,j] = \begin{cases} 1 & (v_i, v_j) \in E \\ 0 & (v_i, v_j) \notin E \end{cases}$
  - Αν έχουμε βάρη: $A[i,j] = w(v_i, v_j)$
  - Παράλληλες ακμές: $A[i,j] = \#(v_i, v_j)$
  - Μη-κατευθυνόμενο: συμμετρικός πίνακας.



	1	2	3	4	5	6
1	0	1	1	0	0	0
2	1	0	1	0	0	0
3	1	1	0	1	1	0
4	0	0	1	0	0	1
5	0	0	1	0	0	0
6	0	0	0	1	0	0

Αναπαράσταση Γραφημάτων

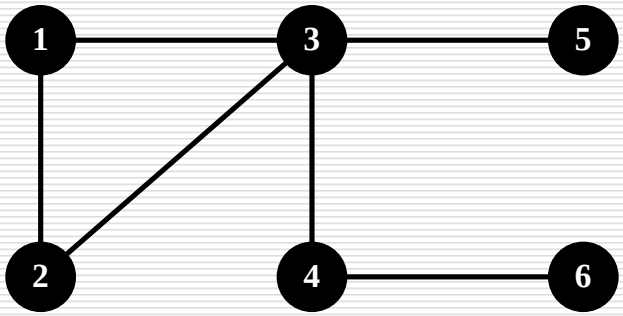
- ... με **πίνακα γειτνίασης:** $A[i,j] = \begin{cases} 1 & (v_i, v_j) \in E \\ 0 & (v_i, v_j) \notin E \end{cases}$
  - Αν έχουμε βάρη, $A[i,j] = w(v_i, v_j)$
  - Μη-κατευθυνόμενο: συμμετρικός πίνακας.
  - Χώρος $\Theta(n^2)$.
  - Άμεσος έλεγχος για ύπαρξη ακμής.
  - Για Πυκνά (Dense) Γραφήματα

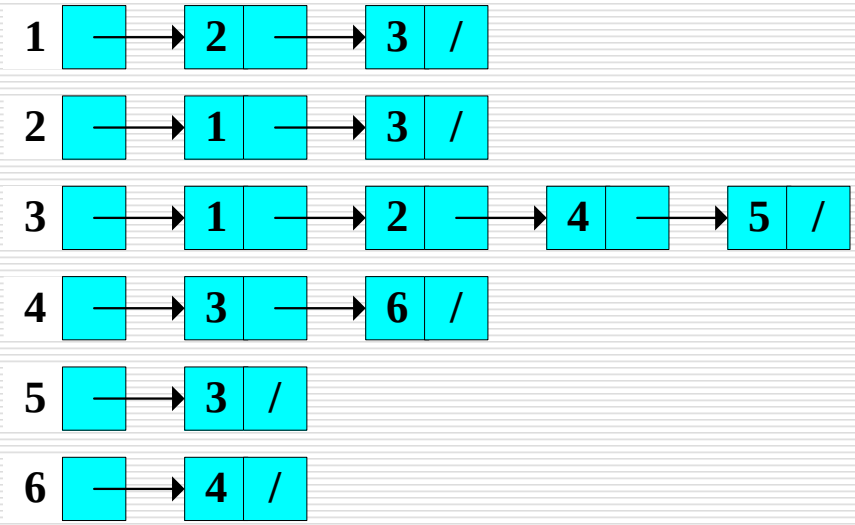


	1	2	3	4	5	6
1	0	0	1	0	0	0
2	1	0	0	1	0	0
3	0	1	0	1	1	0
4	0	1	0	0	0	1
5	0	0	0	0	0	1
6	0	0	1	0	0	0

Αναπαράσταση Γραφημάτων

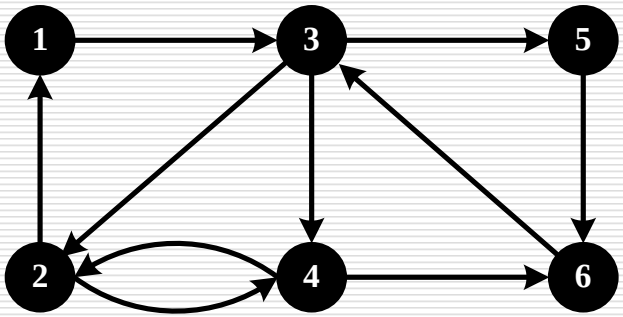
- ... με **λίστα γειτνίασης**: γειτονικές κορυφές σε λίστα.
  - Βάρη αποθηκεύονται στους κόμβους της λίστας.

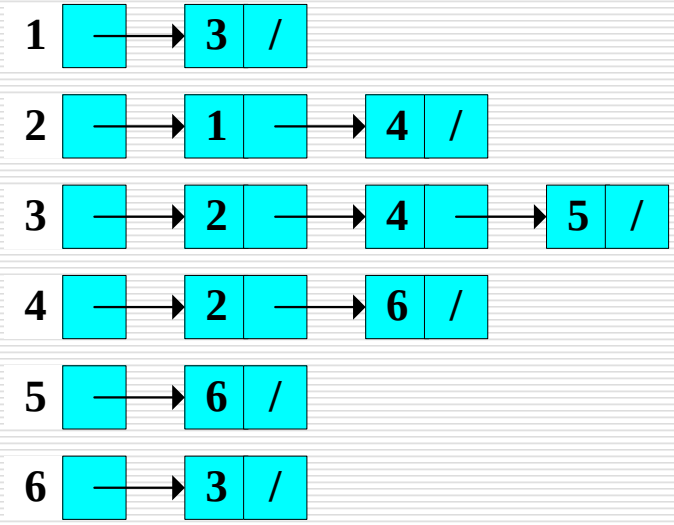




Αναπαράσταση Γραφημάτων

- ... με **λίστα γειτνίασης**: γειτονικές κορυφές σε λίστα.
  - Βάρη αποθηκεύονται στους κόμβους της λίστας.
  - Χώρος Θ(m).
  - Έλεγχος για ύπαρξη ακμής σε χρόνο $O(\deg(v))$.
  - Για Αραιά (Sparse) Γραφήματα





Γενικευμένη Εξερεύνηση

- Συστηματική «επίσκεψη» όλων των κορυφών και ακμών
- Εξαγωγή συμπερασμάτων σχετικά με βασικές ιδιότητες
- Είσοδος: Γράφημα $G = (V, E)$, κορυφή $s \in V$
- Έξοδος: Οι κορυφές που είναι επισκέψιμες από την s
- Βήματα:

1. **Σημείωσε ότι έχεις επισκεφτεί την s**
2. **Όσο υπάρχει $(u, v) \in E$ όπου έχουμε επισκεφτεί την u αλλά όχι την v**
   - I. **Διάλεξε κάποια $(u, v) \in E$**
   - II. **Σημείωσε ότι έχεις επισκεφτεί την v**

- Ανάλογα με τον τρόπο επιλογής της επόμενης ακμής προκύπτει διαφορετικός αλγόριθμος εξερεύνησης

Αναζήτηση Κατά Πλάτος (BFS)

- Εκκίνηση από αρχική κορυφή s και εξέλιξη σε φάσεις.
  - 1^{η} φάση: εξερεύνηση γειτόνων s (σε απόσταση 1 από s).
  - 2^{η} φάση: εξερεύνηση γειτόνων κορυφών $1^{\eta\varsigma}$ φάσης που δεν έχουν εξερευνηθεί ακόμη (σε απόσταση 2 από s).
  - 3^{η} φάση: εξερεύνηση γειτόνων κορυφών $2^{\eta\varsigma}$ φάσης που δεν έχουν εξερευνηθεί ακόμη (σε απόσταση 3 από s).
  - ....
  - k^{η} φάση : εξερεύνηση γειτόνων κορυφών φάσης k – 1 που δεν έχουν εξερευνηθεί ακόμη (σε απόσταση k από s).
- «Κατά Πλάτος»: ολοκληρώνει εξερεύνηση κορυφών σε απόσταση k από s πριν επεκταθεί σε κορυφές σε απόσταση $k+1$.
- Εξέλιξη αναζήτησης: **BFS-δέντρο** (ή δάσος).

Αναζήτηση Κατά Πλάτος (BFS)

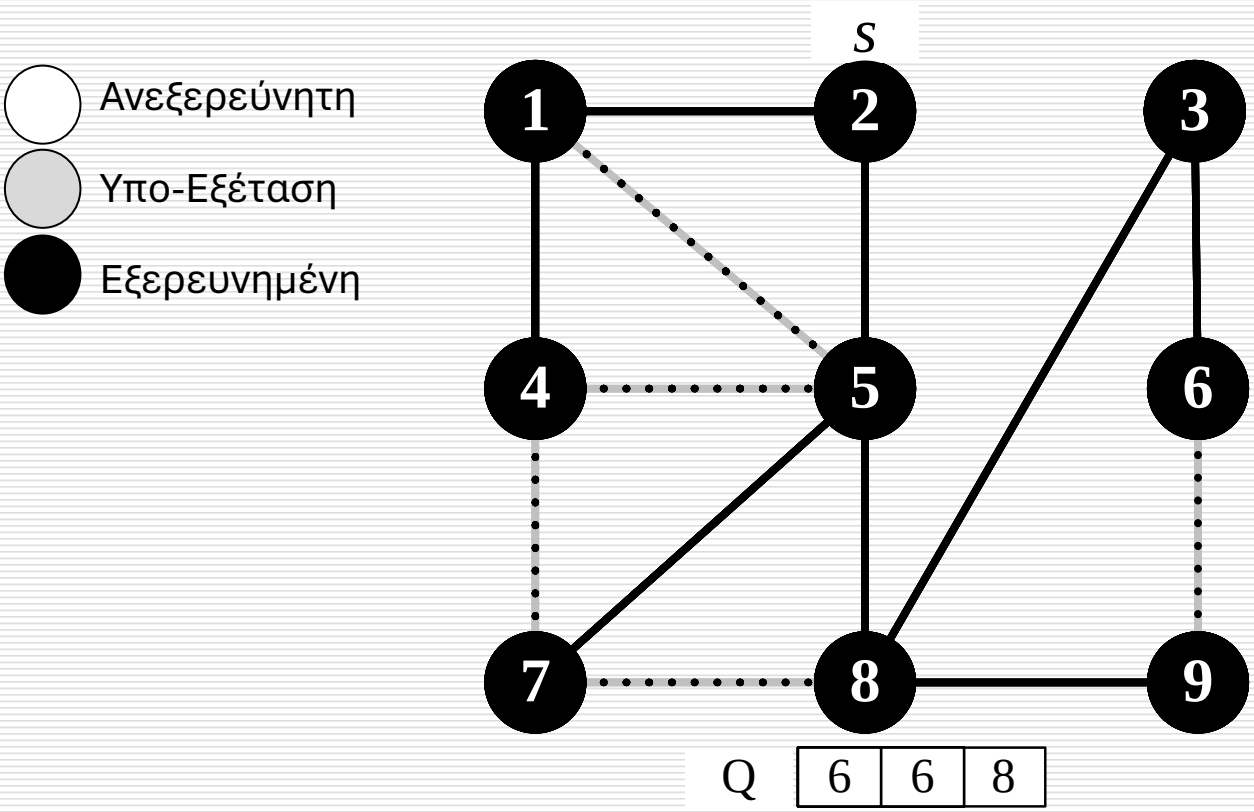
- **Τρία είδη** κορυφών:
  - Ανεξερεύνητη: όχι επίσκεψη ακόμη.
  - Υπο-εξέταση: επίσκεψη αλλά όχι εξερεύνηση γειτόνων.
  - Εξερευνημένη: επίσκεψη και εξερεύνηση γειτόνων.
- Κορυφές περνούν από παραπάνω στάδια με αυτή τη σειρά.
  - Αρχικά όλες οι κορυφές ανεξερεύνητες.
  - Πρώτη επίσκεψη ανεξερεύνητης κορ. → υπό-εξέταση.
  - Επίσκεψη των γειτόνων υπο-εξέταση κορ. → εξερευνημένη.
- «Κατά Πλάτος»:
- σειρά που γίνονται υπο-εξέταση ίδια με σειρά που γίνονται εξερευνημένες.
  - (FIFO) ουρά: εισαγωγή όταν γίνονται υπο-εξέταση (Υ) και εξαγωγή για εξερεύνηση γειτόνων (Ε).

Υλοποίηση

- Πίνακας κατάστασης: $m[v] = \{ A, Y, E \}$.
- Πίνακας γονέων: $p[v]$ = πατέρας v στο BFS-δάσος.

```
BFS(G(V, E), s)
    addToQueue(s); m[s] ← Y; p[s] ← NULL;
    for all v ∈ V \ {s} do
        m[v] ← A; p[v] ← NULL;
    while not emptyQueue() do
        u ← extractFromQueue(); m[u] ← E;
        for all v ∈ L[u] do
            if m[v] = A then
                addToQueue(v); m[v] ← Y; p[v] ← u;
```

Παράδειγμα



Συζήτηση

- Χρόνος εκτέλεσης $\boldsymbol{\Theta(n + m)}$.
- BFS σε (α) πλήρες γράφημα, (β) δέντρο, (γ) κύκλο.
- Ψευδοκώδικας ολοκληρώνεται με κορυφές εξερευνημένες ή ανεξερεύνητες.
  - Αν γράφημα συνεκτικό, όλες εξερευνημένες.
  - Αν όχι, εξερευνημένες σε ίδια συνεκτική συνιστώσα με s. Υπόλοιπες ανεξερεύνητες
  - Τροποποίηση για ολοκλήρωση με όλες εξερευνημένες;

Υπολογισμός Αποστάσεων

```
distances(G,s):
    for all v ∈ V do:
        d[s] <- +∞, m[s] <- A
    d[s] <- 0, m[s] <- E, Q <- {s}
    while not empty(Q) do:
        v <- extract(Q), m[v] <- E
        for all u ∈ L[v] do:
            if m[u] = A then:
                insert(Q,u)
                m[u] = Y
                d[u] = d[v] + 1
    return d
```

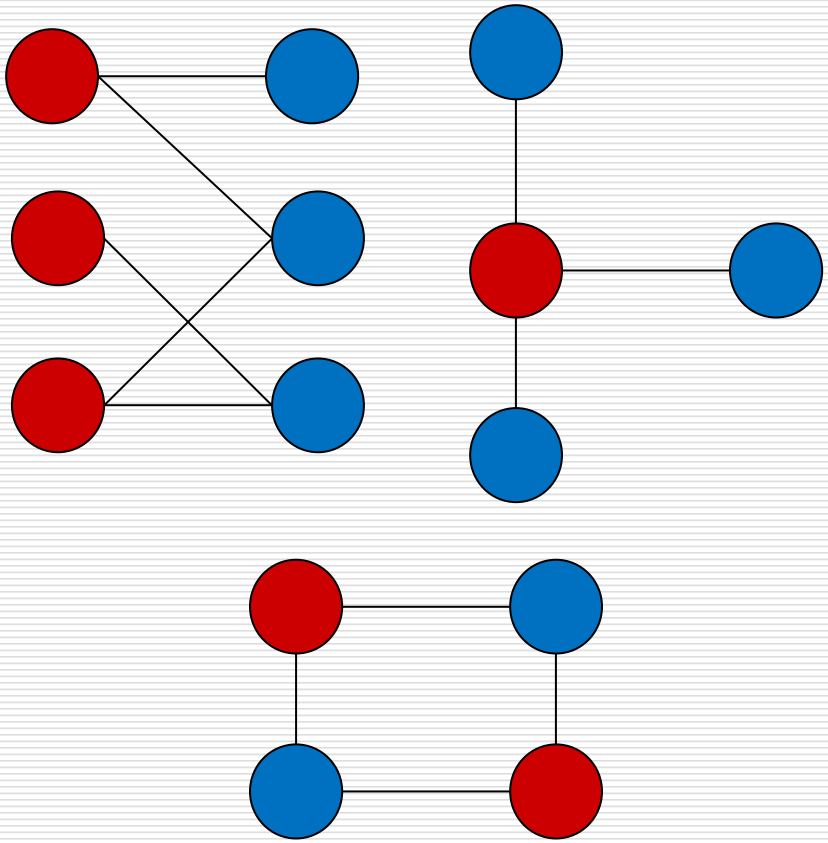
- d: πίνακας αποστάσεων από s
- d[s] = +∞: unreachable
- Είναι οι μικρότερες (shortest paths);

Υπολογισμός connected components

```
Connected_Components(G):
    for all s ∈ V do:
        m[s] <- A
    id_cc <- 0
    for all s ∈ V do:
        if m[s] = A then:
            Q <- {s}, m[s] <- E, id_cc <- id_cc + 1
            while not empty(Q) do:
                v <-extract(Q), m[v] <- E, cc[v] <- id_cc
                for all u ∈ L[v] do:
                    if m[u] = A then:
                        insert(Q,u), m[u] <- Y
    return cc
```

Διμερές Γράφημα

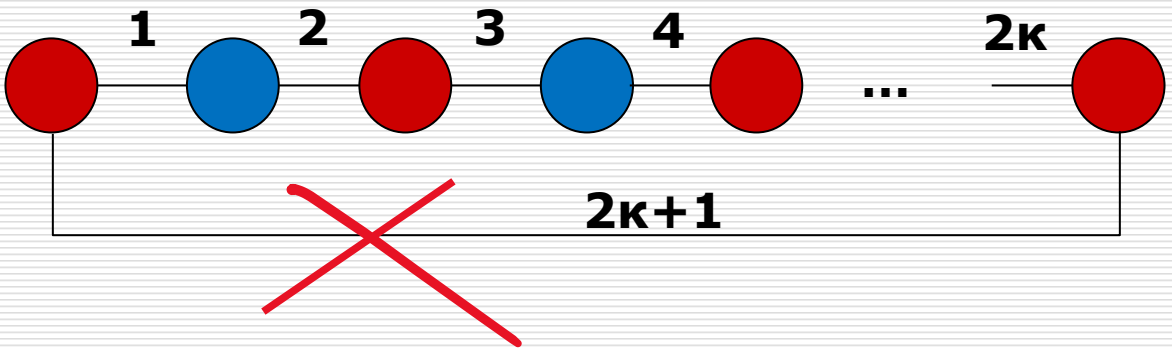
- Υπάρχει διαμέριση του V σε X, Y τέτοια ώστε για κάθε ακμή (a, b):
  - $a \in X, b \in Y$



Ισοδύναμα:
Υπάρχει 2 χρωματισμός ώστε οι κορυφές *κάθε* ακμής να έχουν διαφορετικό χρώμα

Κύκλοι σε διμερή γραφήματα

Αν ένα γράφημα είναι διμερές, τότε ΔΕΝ περιέχει κύκλους με **μονό** μήκος.



- Κόμβοι μονών επιπέδων → Μπλε
- Κόμβοι ζυγών επιπέδων → Κόκκινοι

Έλεγχος Διμέρειας

```
is_BiPartite(G,s):
      Q<-{s}, m[s] <- Y
      for all v e V \ {s} do:
         m[v] <- A, color[v] <- True
      while not empty(Q) do:
         v <- extract(Q), m[v] <- E
         for all u e L[v] do
            if m[u] = A then:
               insert(Q, u), m[u] <- Y
               color[u] <- not color[v]
            elif m[u] = E and color[v] = color[u] then:
               return False
      return True
```

- Εκκίνηση κόκκινο (true)
- Γείτονες μπλε (false)
- Γείτονες γειτόνων κόκκινο
- Αν ακμή με ίδιο χρώμα κορυφών → ΟΧΙ ΔΙΜΕΡΕΣ

Αναζήτηση Κατά Βάθος (DFS)

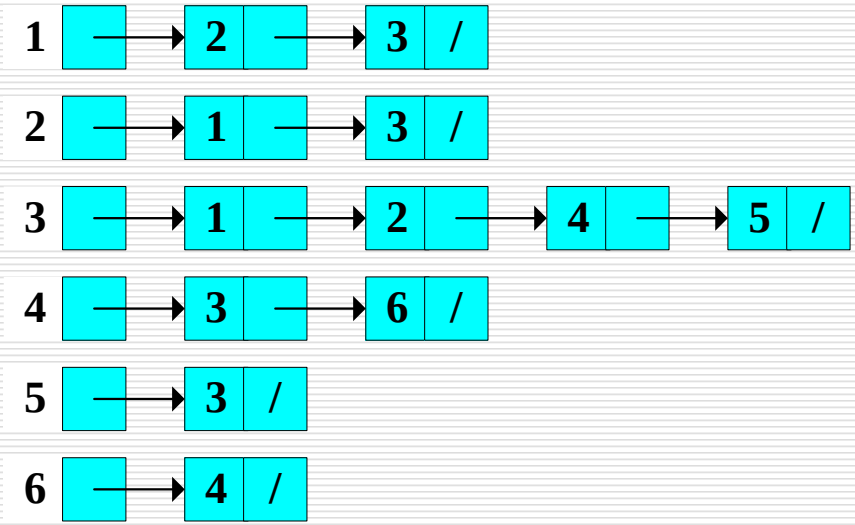
- Εξερεύνηση νέων κορυφών με απομάκρυνση από αρχική.
- Πρώτη επίσκεψη σε ανεξερεύνητη κορυφή u:
  - Εξερεύνηση (αναδρομικά) όλων των (ανεξερεύνητων) γειτόνων της u, πριν ολοκληρώσουμε με u.
- Φύσει αναδρομική διαδικασία:

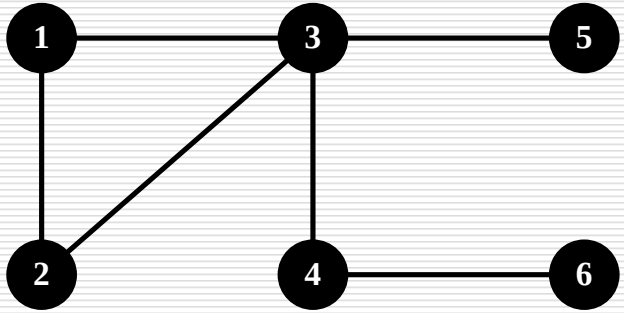
```
DFS(G,u):
  for v ∈ L[u] do:
     if v=A then:
        v <- Y
        DFS(v)
        v <- E
```

- **Τρία είδη** κορυφών:
  - Ανεξερεύνητη (A): όχι επίσκεψη ακόμη – δεν ξέρω ότι υπάρχει
  - Υπο-εξέταση (Y): επίσκεψη αλλά όχι εξάντληση λίστας γειτνίασης
  - Εξερευνημένη (E): εξάντληση λίστας γειτνίασης

Αναζήτηση Κατά Βάθος (DFS)

- Εξερεύνηση νέων κορυφών με απομάκρυνση από αρχική.
- Πρώτη επίσκεψη σε ανεξερεύνητη κορυφή *u*:
  - Εξερεύνηση (αναδρομικά) όλων των (ανεξερεύνητων) γειτόνων της *u*, πριν φύγουμε από την u.





Αναζήτηση Κατά Βάθος (DFS)

- Κορυφές περνούν από παραπάνω στάδια:
  - Αρχικά όλες οι κορυφές ανεξερεύνητες.
  - Πρώτη επίσκεψη ανεξερεύνητης κορ. → υπό-εξέταση.
  - Ολοκλήρωση DFS για (ανεξερ.) γείτονες κορ. → εξερευνημένη.
- Κορυφή u τίθεται υπό-εξέταση:
  - Όλες οι ανεξερεύνητες κορυφές που είναι προσπελάσιμες από u και θα τεθούν εξερευνημένες πριν u τεθεί εξερευνημένη.
  - Δηλαδή: $Σειρά(A \rightarrow Y) \neq Σειρά(Y \rightarrow E)$ $(LIFO)$
  - Ενώ σε BFS: $Σειρά(A \rightarrow Y) = Σειρά(Y \rightarrow E)$ $(FIFO)$
- Εξέλιξη διαδικασίας αποτυπώνεται σε **DFS-δάσος** και «**χρόνους**» πρώτης επίσκεψης (Y) και αναχώρησης (E).
  - DFS-δάσος: ακμές πρώτης επίσκεψης, ακυκλικό.

Υλοποίηση

- Πίνακας κατάστασης: **m[v]** = { A, Y, E }.
- Πίνακας γονέων: **p[v]** = πατέρας v στο DFS-δάσος.
- «Χρόνοι» πρώτης επίσκεψης **d[v]** και αναχώρησης **f[v]**.

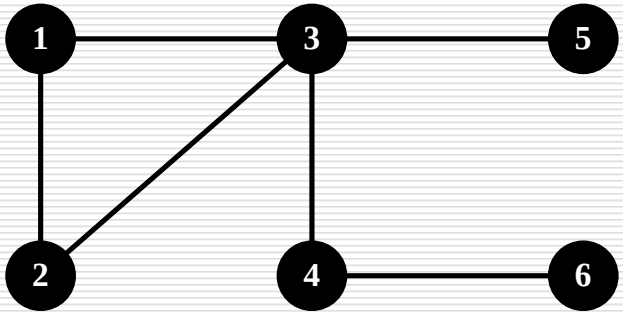
```
DFS_Init(G(V, E))
    t ← 0;
    for all v ∈ V do
        m[v] ← A; p[v] ← NULL;
    for all v ∈ V do
        if m[v] = A then DFS(v);
```

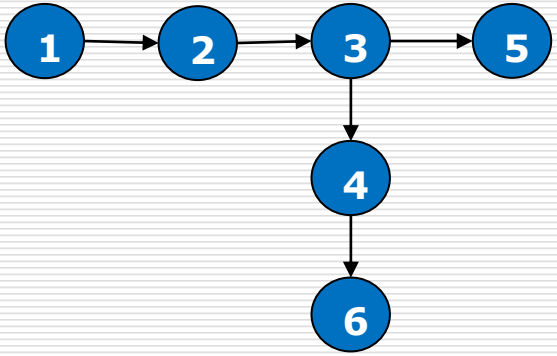
```
DFS(v)
    m[v] ← Y; d[v] ← ++t;
    for all u ∈ L[v] do
        if m[u] = A then
            p[u] ← v; DFS(u);
    m[v] ← E; f[v] ← ++t;
```

- Χρόνος εκτέλεσης **Θ(n + m)**.
- DFS σε (α) δέντρο, (β) πλήρες γράφημα, (γ) κύκλο.

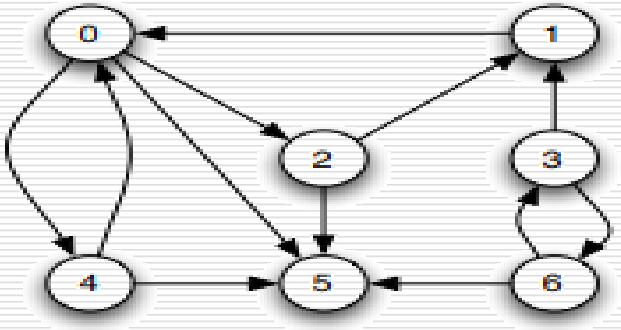
Παραδείγματα DFS δάσους

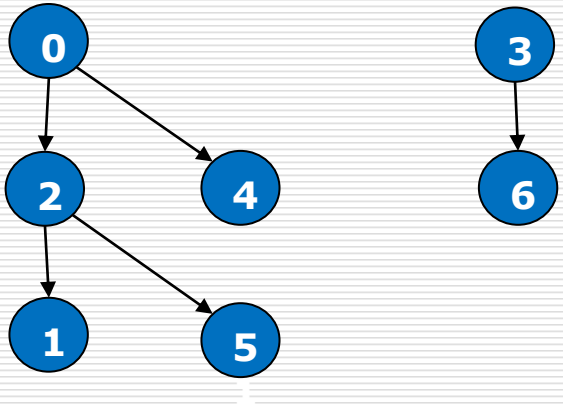
- Μη κατευθυνόμενο





- Κατευθυνόμενο

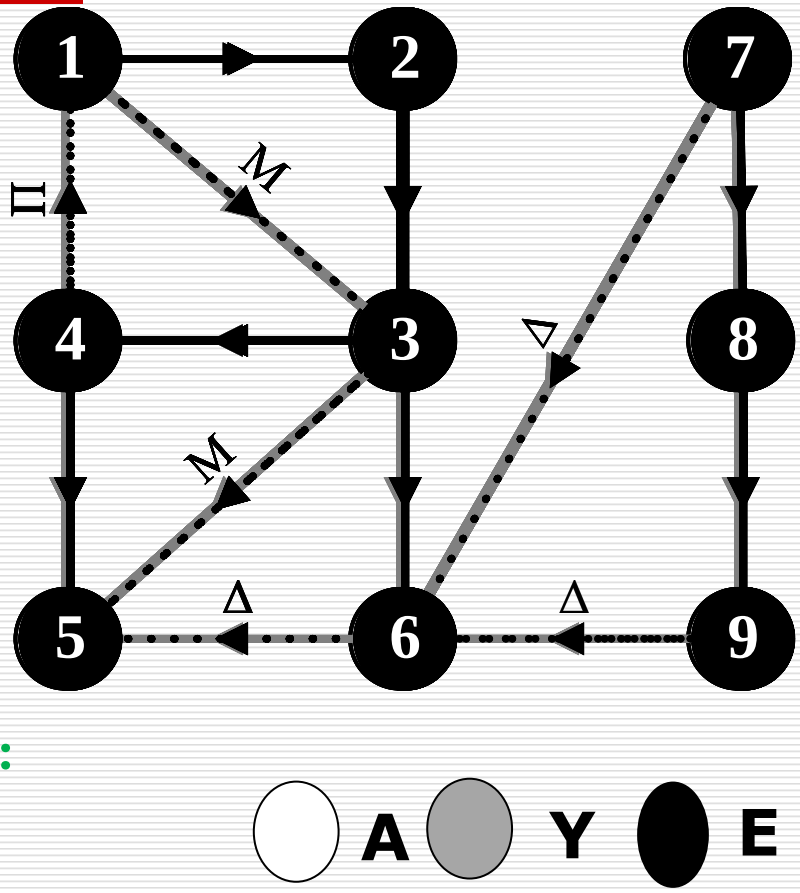




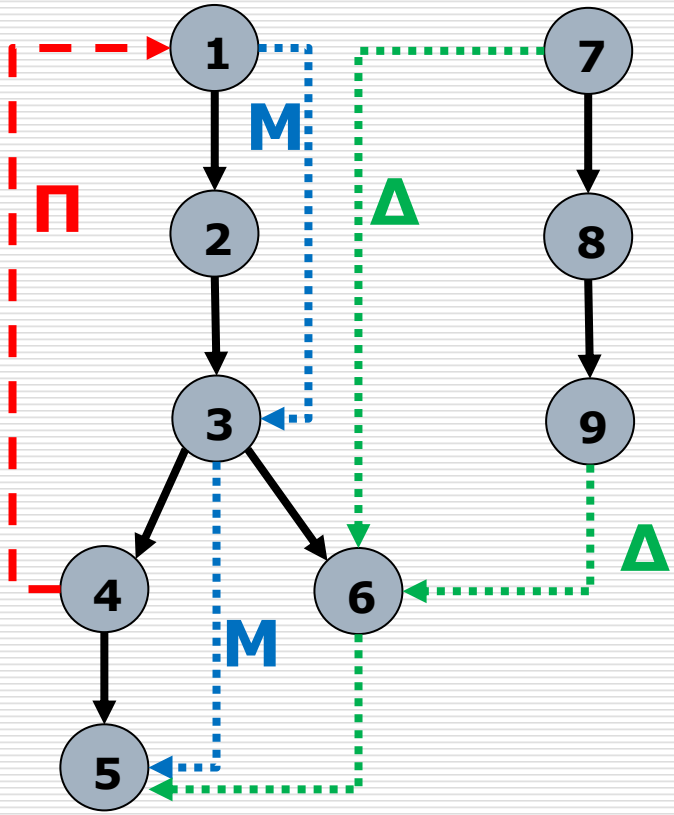
Παράδειγμα – Κατηγορίες Ακμών

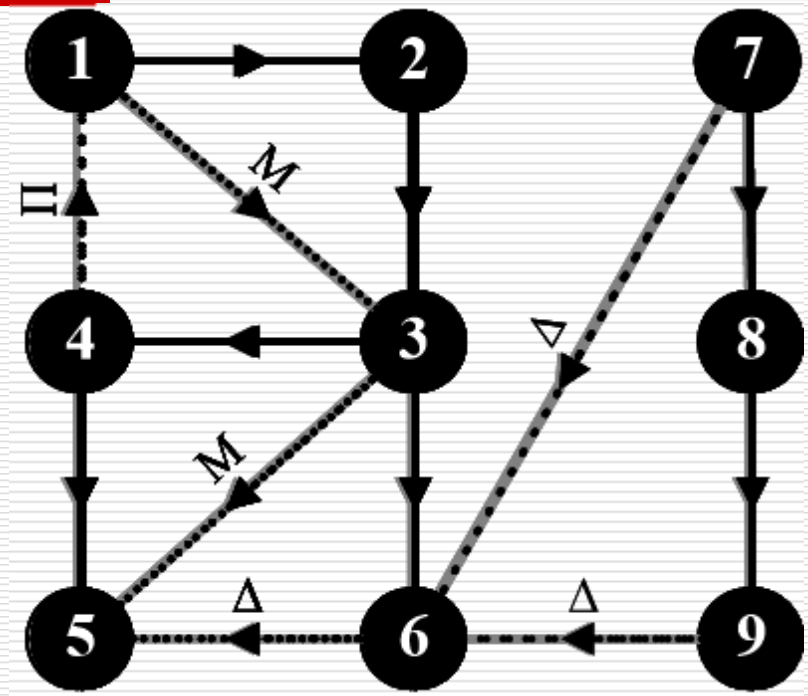
Κατηγορία Ακμής *(u,v)*:

- u υπό-εξέταση
- **Ακμή Δάσους / δέντρου:**
  - Όταν v ανεξερεύνητη.
- **Πίσω ακμή (ανιούσα):**
  - Όταν v υπό-εξέταση
  - Υποδηλώνει κύκλο.
- **Μπροστά ακμή (κατιούσα):**
  - Όταν v εξερευνημένη *και v απόγονος u στο δέντρο*.
- **Ακμή διασταύρωσης (εγκάρσιες):**
  - όταν v εξερευνημένη *και v **όχι** απόγονος u στο δέντρο*.



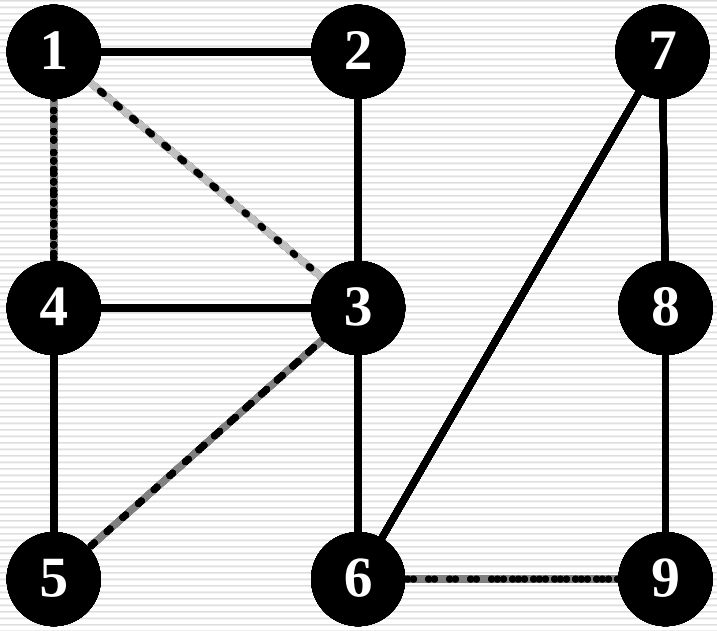
Παράδειγμα – Κατηγορίες Ακμών (2)



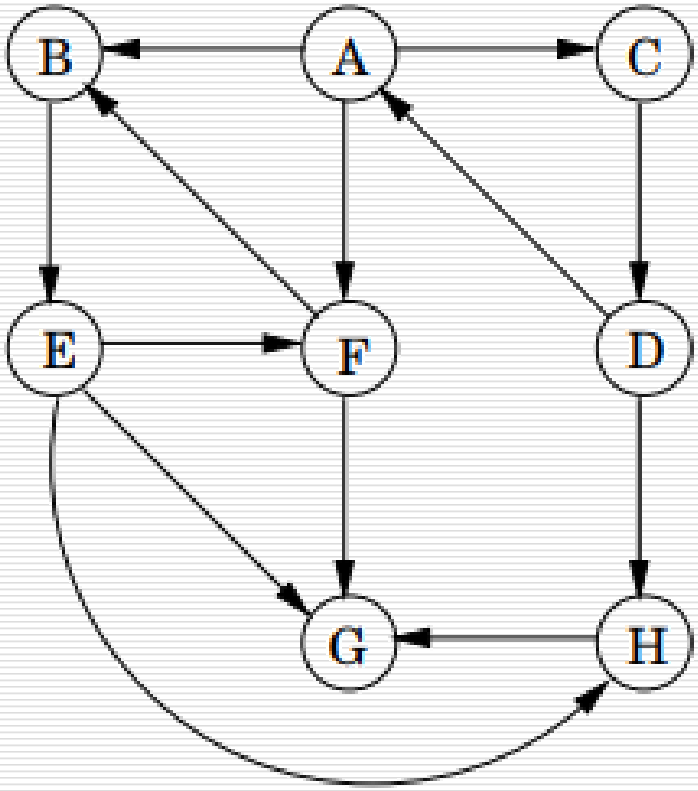


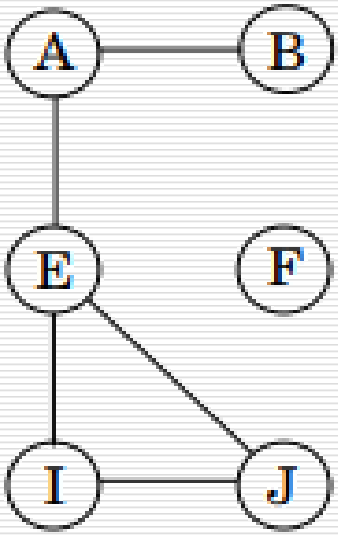
Σε μη-κατευθυνόμενο γράφημα

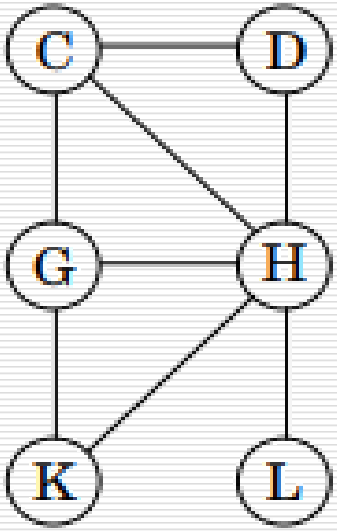
- DFS παράγει μόνο ακμές δέντρου και πίσω ακμές.
  - Ακμή {v, u} με d[v] < d[u] (πρώτα πρώτη επίσκεψη σε v).
  - Πρώτα v Y, μετά u Y, μετά u E, τέλος v E.
  - Αν κατεύθυνση v → u εξερευνήθηκε πρώτη, τότε {v, u} ακμή δέντρου.
  - Αν κατεύθυνση u → v εξερευνήθηκε πρώτη, τότε {v, u} πίσω ακμή.



Εξάσκηση







Μερικές Ιδιότητες

Γράφημα ακυκλικό **αν** DFS δεν παράγει πίσω ακμές.

ΕΥΘΥ: Ύπαρξη πίσω ακμής ⇒ Κύκλος (αντιθετοαντίστροφο)
Εξερεύνηση πίσω ακμής (u, v) όταν v Y ⇒ Μονοπάτι v → u ⇒ Μονοπάτι v → u και ακμή (u, v) ⇒ κύκλος.

ΑΝΤΙΣΤΡΟΦΟ: Κύκλος ⇒ Πίσω ακμή (αντιθετοαντίστροφο)
Έστω κύκλος C, v πρώτη κορυφή C που τίθεται Y, και (u, v) ακμή C που εισέρχεται στην v.
u απόγονος της v στο DFS-δάσος γιατί:

- Υπάρχει v → u μονοπάτι.
- Όλες οι άλλες κορυφές του C είναι A όταν v γίνεται Y.

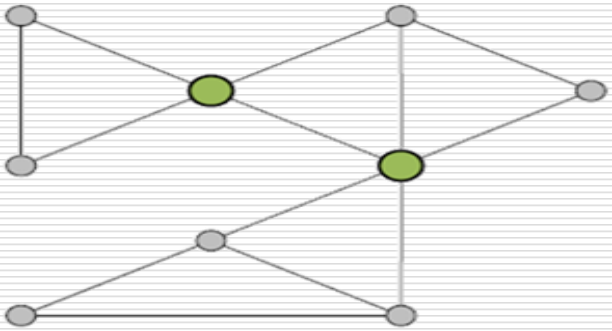
Άρα (u, v) πίσω ακμή.

Μερικές Ιδιότητες

- Για μη-κατευθυνόμενα γραφήματα, DFS υπολογίζει συνεκτικές συνιστώσες (όπως και BFS).
- Αν v απόγονος u στο DFS-δάσος, $[d[v], f[v]] \subset [d[u], f[u]]$
  Αν v όχι απόγονος u στο DFS-δάσος, $[d[v], f[v]] \cap [d[u], f[u]] = \emptyset$
- «Χρόνοι» πρώτης επίσκεψης και αναχώρησης δίνουν πληροφορίες για δομή γραφήματος:
  - Σημεία κοπής και γέφυρες σε μη-κατευθυνόμενα γραφήματα.
  - Τοπολογική διάταξη σε Directed Acyclic Graphs (DAGs).
  - Ισχυρά συνεκτικές συνιστώσες σε κατευθυνόμενα γραφήματα.

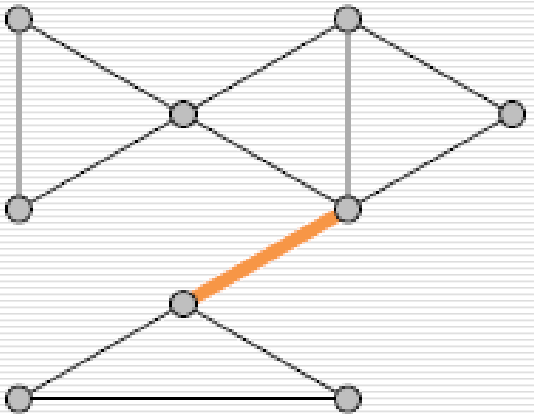
Σημεία Κοπής – Γέφυρες (Μη-Κατευθυνόμενα)

- Κορυφή v - Σημείο κοπής (cut vertex, articulation point):
  - αφαίρεση v → αυξάνει πλήθος συνεκτικών συνιστωσών.



  - Brute Force: $O(n(n+m))$

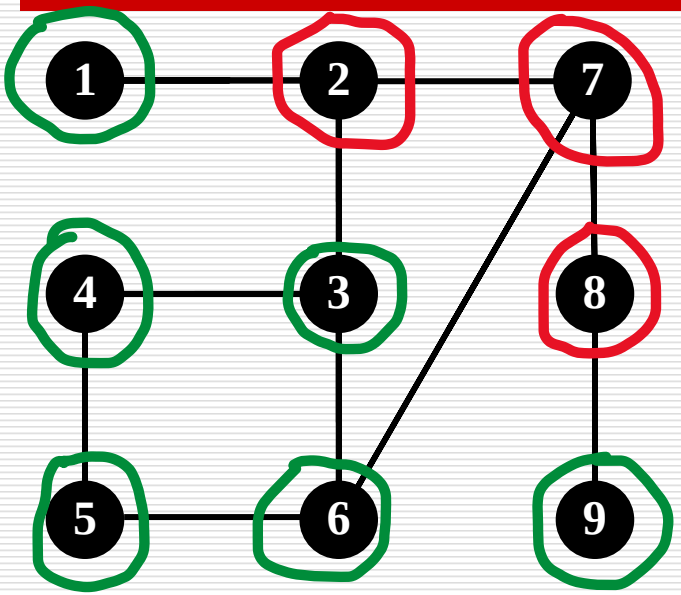
- Ακμή e -Γέφυρα (bridge):
  - αφαίρεση e αυξάνει πλήθος συνεκτικών συνιστωσών

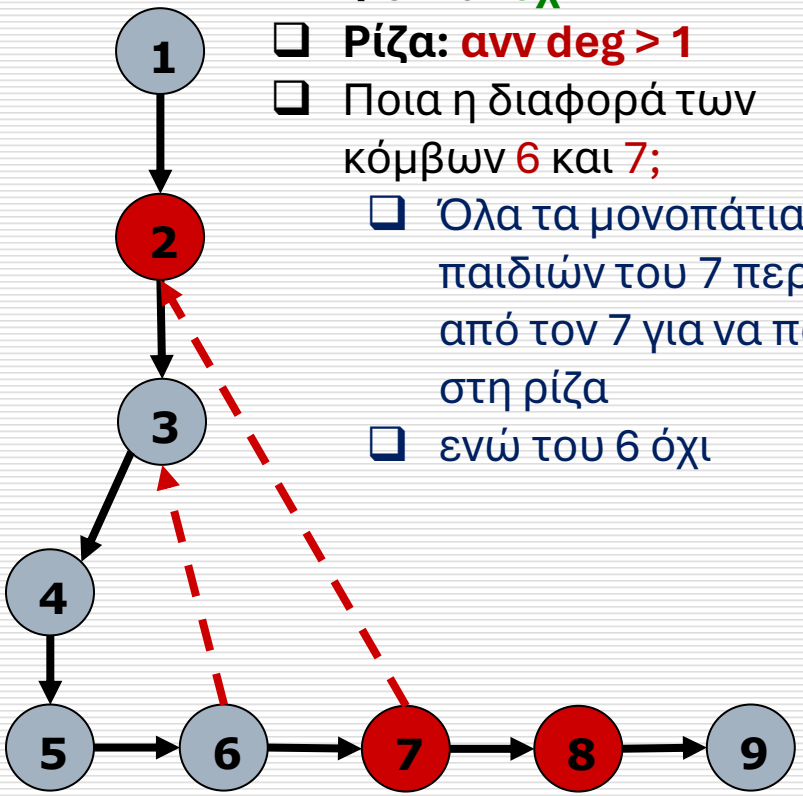


  - Brute Force: $O(m(n+m))$

Εύρεση σε γραμμικό χρόνο με DFS

Σημεία κοπής με DFS (γραφικά)

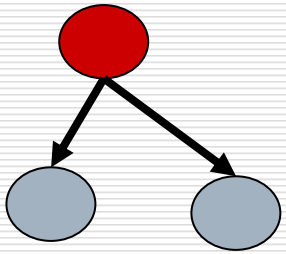


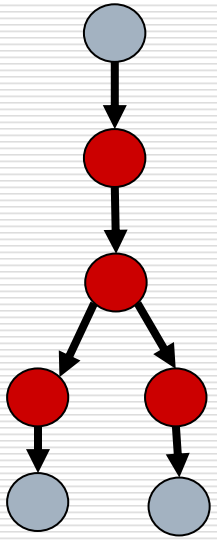


- **Φύλλα: όχι**
- **Ρίζα: ανν deg > 1**
- Ποια η διαφορά των κόμβων 6 και 7;
  - Όλα τα μονοπάτια των παιδιών του 7 περνούν από τον 7 για να πάνε στη ρίζα
  - ενώ του 6 όχι

Σημεία Κοπής

- Ρίζα r DFS-δέντρου T είναι σημείο κοπής ανν:
  - r έχει 2 ή περισσότερα παιδιά στο T.
  - Δεν υπάρχουν ακμές διασταύρωσης: όλα τα μονοπάτια από ένα υποδέντρο στο άλλο διέρχονται μέσω r.
- Κορυφή v $(\neq r)$ είναι σημείο κοπής ανν
  - **υπάρχει** απόγονος u της v στο T τ.ω. όλα $u - r$ μονοπάτια στο G διέρχονται από v.
- Κορυφή v $(\neq r)$ **δεν** είναι σημείο κοπής ανν:
  - **κάθε** απόγονος u της v στο T μπορεί να «παρακάμψει» v (μέσω πίσω ακμής).
  - … ανν από κάθε υποδέντρο με ρίζα παιδί της v έχει πίσω ακμή που καταλήγει σε πρόγονο της v.





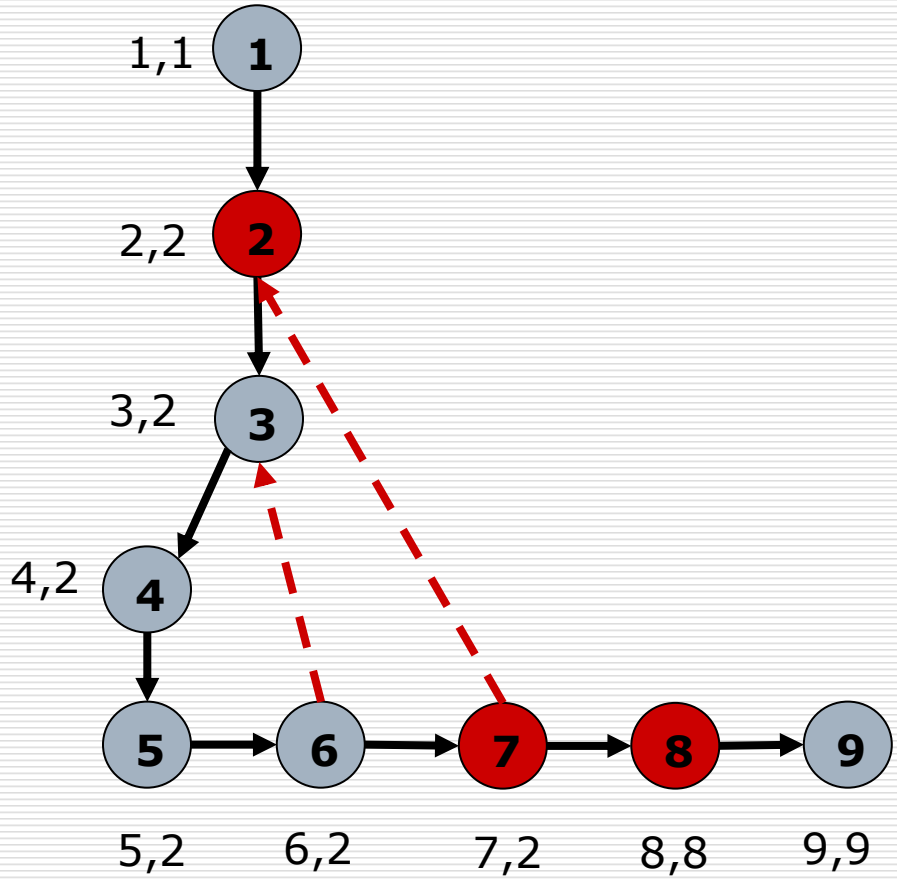
Σημεία Κοπής – Υπολογισμός low

- Πόσο ψηλά μπορείς να φθάσεις με πίσω ακμή

$$low[v] = min \begin{cases} d[v] \\ d[w] \end{cases}$$

- όπου (u, w) πίσω ακμή (απογόνου u της v)

v (μη ρίζα) σημείο κοπής:
Για κάθε παιδί u της v: $d[v] \leq low[u]$



DFS με σημεία κοπής

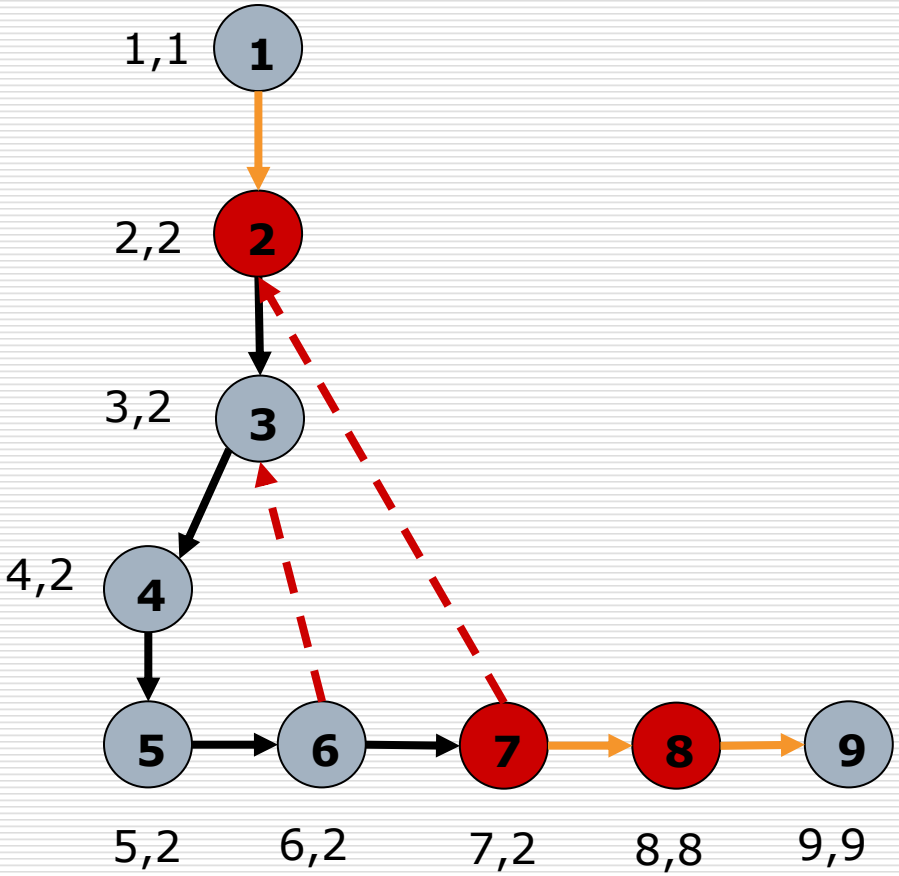
```
DFS(u):
        t <- t+1, d[v] <- t, m[v] <- Y
        low[v] <- t, isAP[v]  = False, children <- 0
        for u ∈ L[v] do:
                if m[u] = A then: #ακμή δέντρου DFS
                        p[u] <- v, DFS(u)
                        low[v] <- min(low[u],low[v])
                        if p[v]=NULL then:
                           isAP[v] <- children > 1
                        else:
                           isAP[v] <- (d[v] <= low[u])
                        elif not u = p[v]: #πίσω ακμή
                           low[v] = min(low[v],low[u])
        m[u] = E
```

Γέφυρες

- Ακμή e (μη κατευθυνόμενου γραφήματος) είναι γέφυρα αν αφαίρεση e αυξάνει πλήθος συνεκτικών συνιστωσών
- Ακμή $e = \{v, u\}$

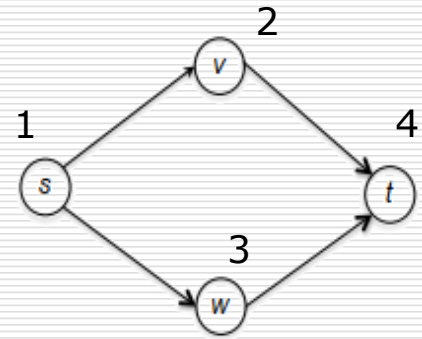
(v πατέρας u στο DFS δέντρο) είναι γέφυρα ανν

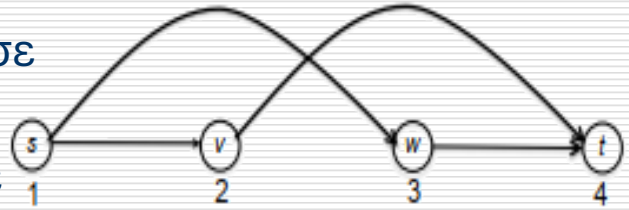
$$low(u) > d(v).$$



DAGs (Directed Acyclic Graphs)

- DAG (Directed Acyclic Graph)
  - Source: $in - degree = 0$
  - Sink: $out - degree = 0$
- Αντιστοιχεί σε σχέση μερικής διάταξης:
  - Ακμή (u, v) ⇔ δ(u) ≤ δ(v) (δηλ. u «προηγείται» v).
- Εφαρμογές DAG
  - Σειρά υπολογισμού αριθμητικών παραστάσεων σε compilers.
  - Προγραμματισμός εργασιών σε σύνθετα έργα με προτεραιότητες.
  - Επίλυση γραμμικών συστημάτων ανισώσεων (περιορισμών)

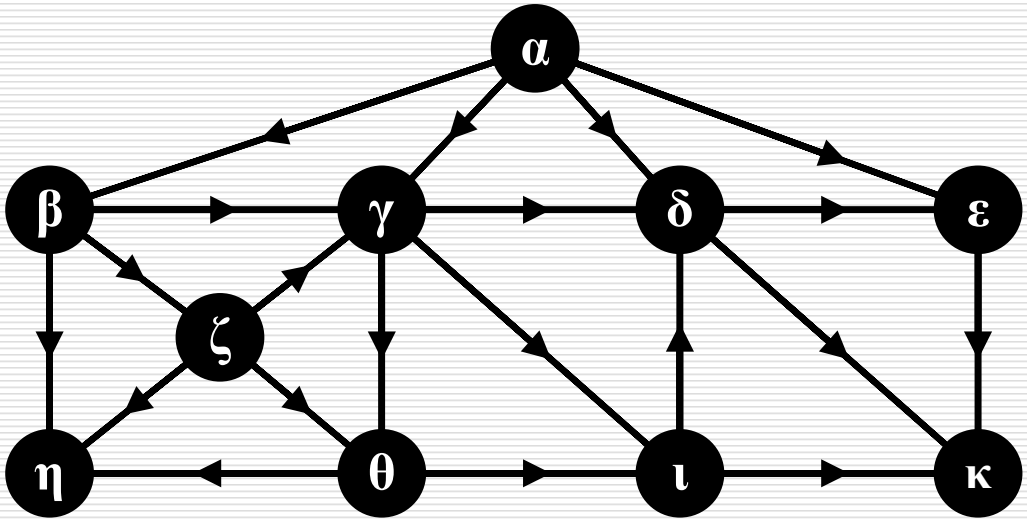




Τοπολογική Ταξινόμηση

- Τοπολογική διάταξη ανν γράφημα ακυκλικό (DAG).
  - **Ευθύ:**
    - Ύπαρξη κύκλου ασύμβατο με διάταξη
  - **Αντίστροφο:**
    - Αφαίρεση **sources** του DAG και προσκείμενες ακμές
- **DFS** ελέγχει για ύπαρξη κύκλων και υπολογίζει «σειρά» κορυφών **συμβατή** με μερική διάταξη του DAG
- Τοπολογική διάταξη: Κορυφές σε φθίνουσα σειρά χρόνων αναχώρησης του DFS, δηλ. $f[v_1] > f]v_2] > \cdots > f[v_n]$
  - Χρόνος $\Theta(n+m)$.

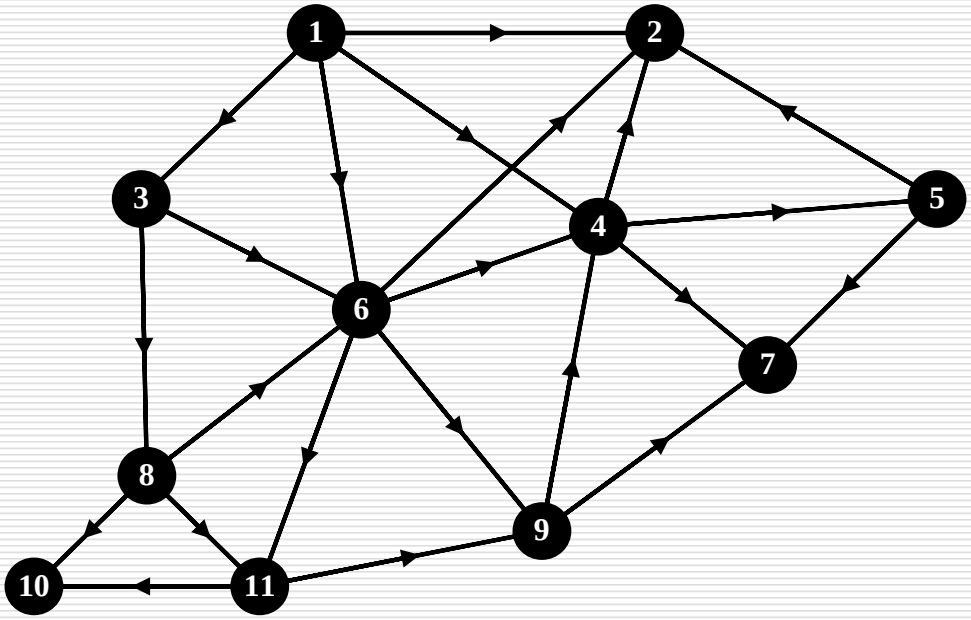
Παράδειγμα 1



v	d	f
α	1	**20**
β	2	**19**
γ	3	**16**
δ	4	**9**
ε	5	**8**
ζ	17	**18**
η	11	**12**
θ	10	**15**
ι	13	**14**
κ	6	**7**

Τοπολογική διάταξη:
α, β, ζ, γ, θ, ι, η, δ, ε, κ

Παράδειγμα 2 - άσκηση



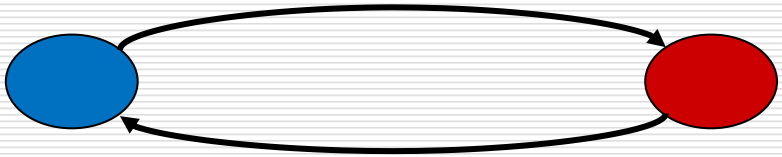
Τοπολογική διάταξη:
1, 3, 8, 6, 11, 10,
9, 4, 5, 7, 2

Τοπολογική Ταξινόμηση: Ορθότητα

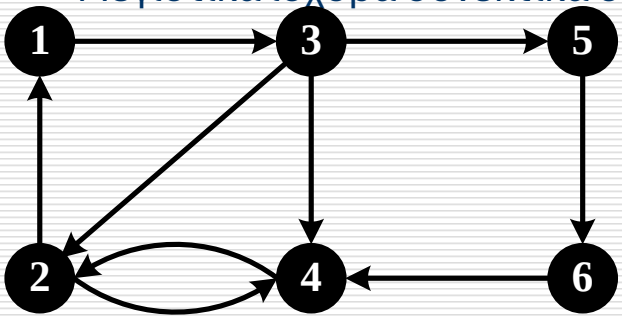
- Έστω DAG G(V, E). Θδο $\forall (u, v) \in E, f[u] > f[v]$.
  - Εξερεύνηση (u, v) σημαίνει u = Y και $v \in \{A, E, Y\}$:
  - **v = A**
    - v απόγονος της u στο DFS-δάσος.
    - Άρα f[u] > f[v], γιατί πρώτα τίθεται f[v] και μετά f[u].
  - **v = E**
    - εξερεύνηση της v ολοκληρώθηκε πριν ολοκληρωθεί εξερεύνηση u, άρα f[u] > f[v].
  - **v = Y**
    - αποκλείεται γιατί σημαίνει πίσω ακμή δηλ. κύκλο.

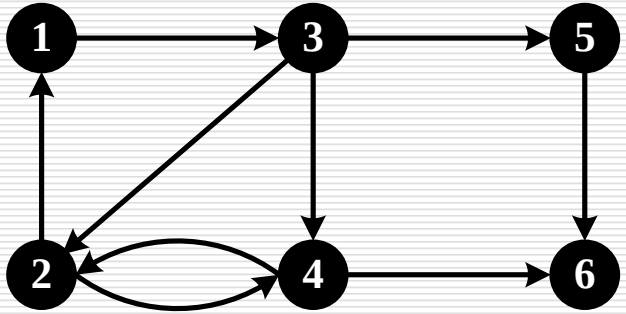
Ισχυρά Συνεκτικές Συνιστώσες

- Κατευθυνόμενο γράφημα G(V, E) **ισχυρά συνεκτικό** αν
  $\forall u, v \in V$, υπάρχουν u – v και v – u μονοπάτια.
  - Για κάθε ζευγάρι κορυφών ισχυρά συνεκτικού γραφήματος, υπάρχει κυκλική διαδρομή που τις περιλαμβάνει.



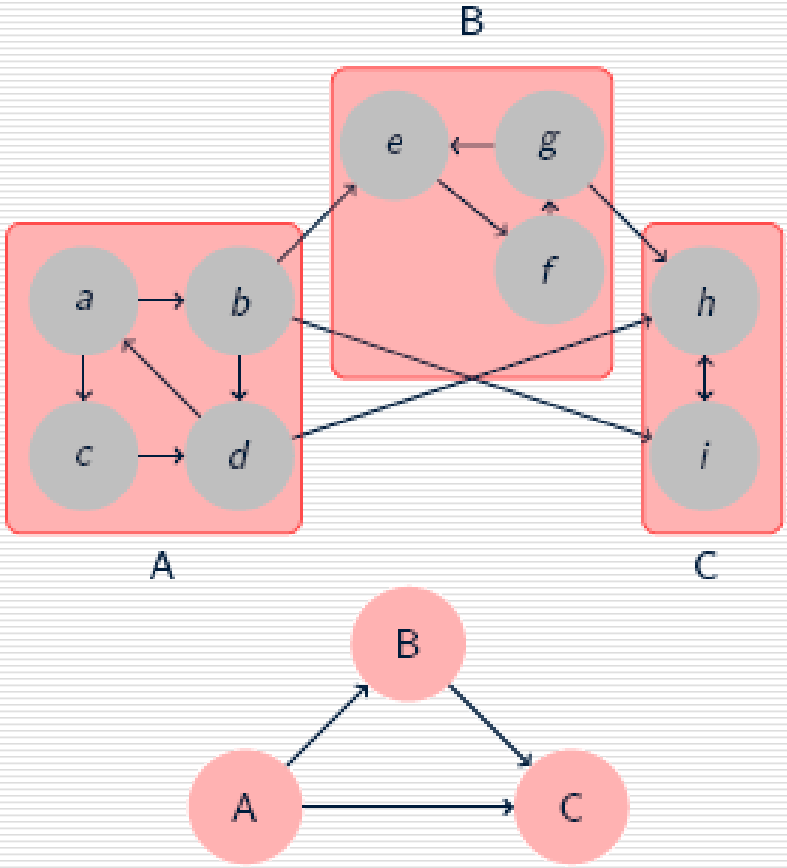
- Αν ένα κατευθυνόμενο γράφημα δεν είναι ισχυρά συνεκτικό, διαμερίζεται σε ισχυρά συνεκτικές συνιστώσες:
  - Μεγιστικά ισχυρά συνεκτικά υπογραφήματα.





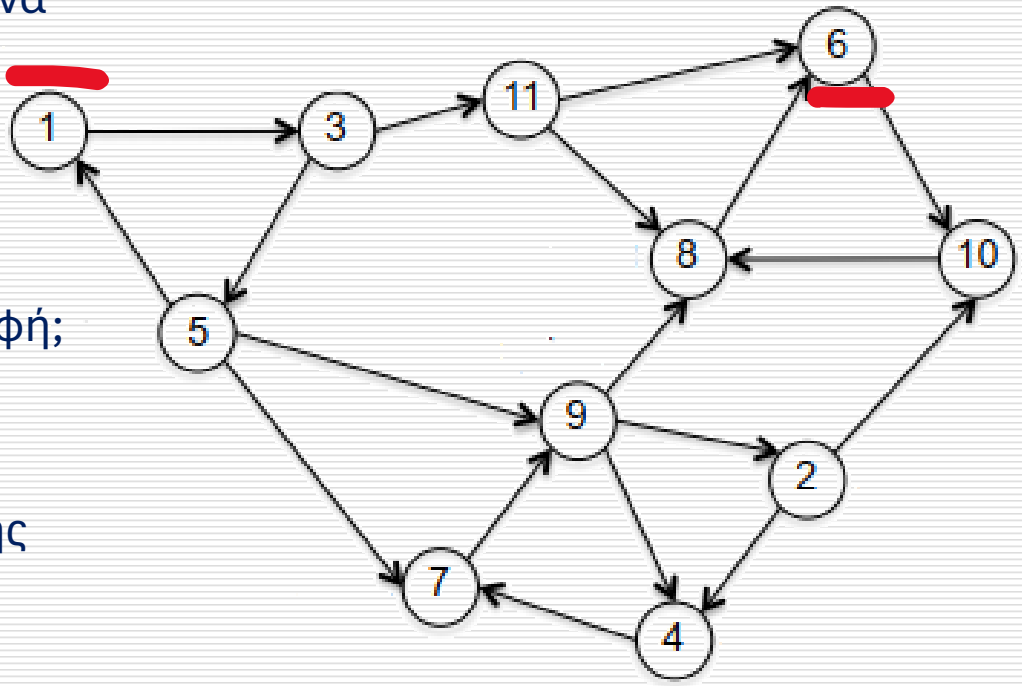
Ισχυρά Συνεκτικές Συνιστώσες (Μεταγράφημα)

- (Κατευθυνόμενο) γράφημα ισχυρά συνεκτικών συνιστωσών (ΙΣΣ):
  - Κορυφή για κάθε ΙΣΣ
  - Ακμή (X, Y) από ΙΣΣ X προς ΙΣΣ Y ανν (αρχικά) υπάρχει ακμή (v, u) για $v \in X$ και $u \in Y$.
- Γράφημα ΙΣΣ είναι **ακυκλικό** (DAG) και μπορεί να ταξινομηθεί τοπολογικά.
- Μπορούμε να βρούμε ΙΣΣ με DFS?
  - **ΝΑΙ**, αν ξεκινήσουμε από την κατάλληλη κορυφή
  - **Πώς** θα τη βρούμε?

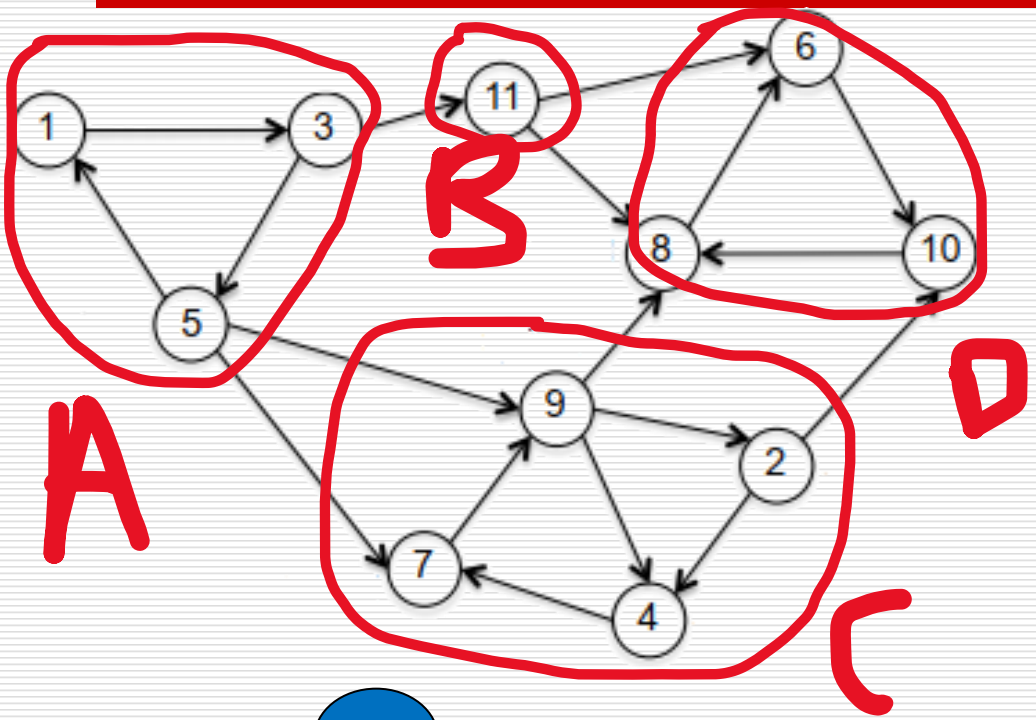


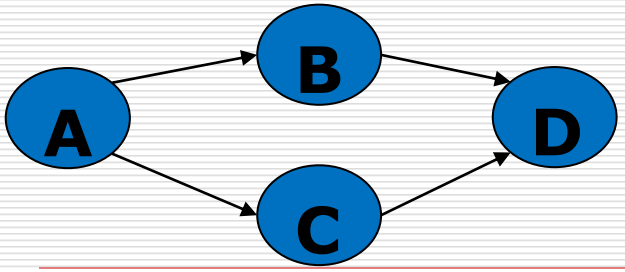
Υπολογισμός Ισχυρά Συνεκτικών Συνιστωσών

- **Διαίσθηση:**
  - Θέλουμε κάποια κορυφή που να ανήκει σε sink ΙΣΣ
- **Ιδέα:**
  - Υπολογισμός τοπολογικής διάταξης (χωρίς πίσω ακμές)
  - Εκκίνηση από τελευταία κορυφή;
- **Πρόβλημα:**
  - **Δεν** είναι υποχρεωτικό ότι τελευταία κορυφή τοπολογικής διάταξης ανήκει σε sink ΙΣΣ



Υπολογισμός Ισχυρά Συνεκτικών Συνιστωσών





(μία) Τοπολογική Ταξινόμηση:

- **1**
- 3
- **11**
- 5
- **7**
- 9
- 2
- 10
- 8
- **6**
- *4 (ανήκει C και όχι D)*

Υπολογισμός Ισχυρά Συνεκτικών Συνιστωσών

- Όμως ισχύει το *ανάστροφο*:
  - Το πρώτο στοιχείο της τοπολογικής ταξινόμησης ανήκει στο source ΙΣΣ
- Και κάτι ισχυρότερο:
  - Αν ονομάσω κάθε ΙΣΣ με την μικρότερη κορυφή του στην τοπολογικής ταξινόμησης λαμβάνω τοπολογική ταξινόμηση του μεταγραφήματος των ΙΣΣ
  - Απόδειξη
    - Βλ. ορθότητα τοπολογικής ταξινόμησης
- Πώς θα λύσω το πρόβλημα με το sink?
  - Υπολογισμός ανάστροφου γραφήματος ώστε να βρεθεί στο source

Αλγόριθμος Kosaraju (– Sharir)

Εύρεση ΙΣΣ σε σε κατευθυνόμενο G(V, E):

- **Υπολόγισε** τοπολογική διάταξη με DFS (αγνοώντας τις πίσω ακμές).
- **Υπολόγισε** το ανάστροφο γράφημα G^T (αντιστροφή κατεύθυνσης ακμών, G και G^T έχουν ίδιες ΙΣΣ).
- **Εφάρμοσε** DFS σε G^T με σειρά της τοπολογικής διάταξης.
  - Κάθε φορά που φτάνουμε σε αδιέξοδο: νέα ΙΣΣ.

Δύο DFS: $\Theta(n+m)$

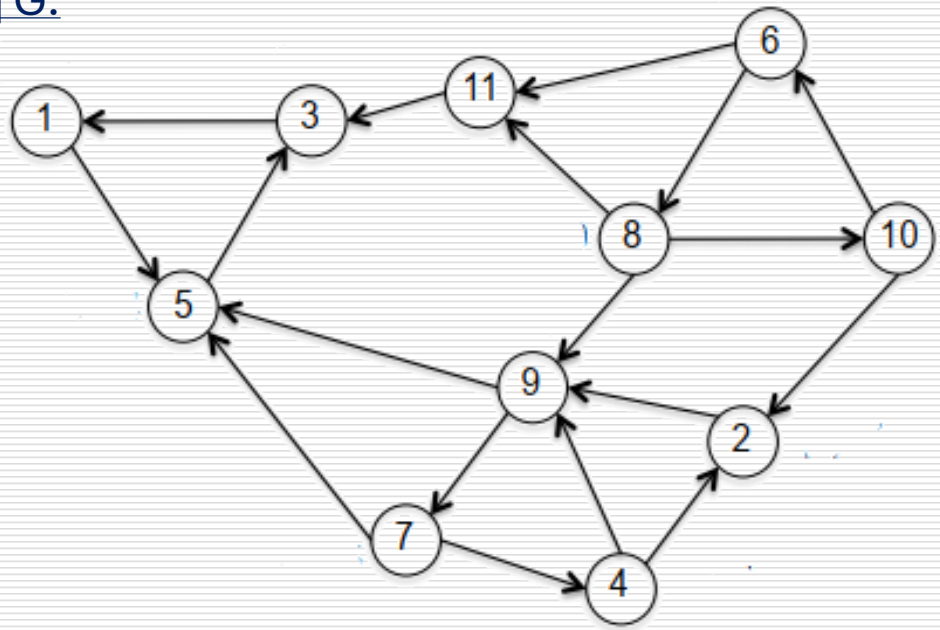
Βελτίωση: Tarjan με μία DFS.

Ολοκληρωμένο παράδειγμα

Τοπολογική Ταξινόμηση G:

- 1
- 3
- 11
- 5
- 7
- 9
- 2
- 10
- 8
- 6
- 4

Υπολογισμός G^T



DFS(G^T) με σειρά τοπολογικής

- 1 → {1,3,5}
- 3 → E
- 11 → {11}
- 5 → E
- 7 → {7,4,2,9}
- 9 → E
- 2 → E
- 10 → {10,8,6}
- 8 → E
- 6 → E
- 4 → E