



Προγραμματιστικές Τεχνικές

Γράφοι: το πρόβλημα της εξερεύνησης

Εξερεύνηση κατά Πλάτος (BFS) και κατά Βάθος (DFS)

Εφαρμογές: συντομότερες διαδρομές, τοπολογική διάταξη

Διδάσκοντες

Γιώργος Αλεξανδρίδης

Βασίλης Βεσκούκης

Θάνος Βουλόδημος

Γιώργος Γκούμας

Άρης Παγουρτζής

Γιώργος Σιόλας

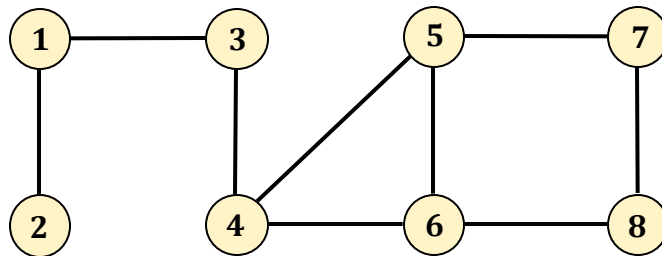
Γιώργος Στάμου

Παρασκευή Τζούβελη

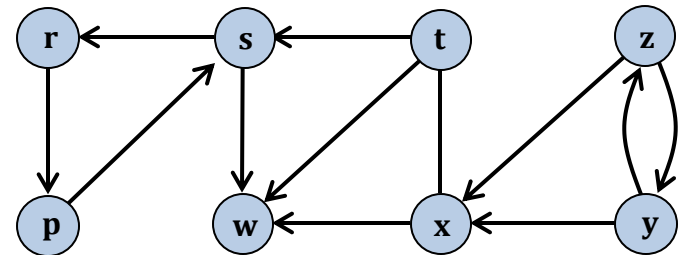
progtech@cslab.ece.ntua.gr

Ορισμός Γράφων - Δικτύων

Γράφος - Δίκτυο



$$G_1 = (V_1, E_1)$$

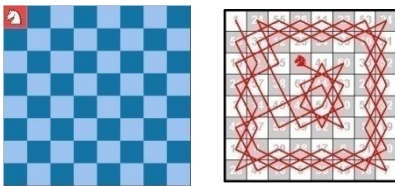


$$G_2 = (V_2, E_2)$$

- **Γράφος (ή γράφημα) :** ζεύγος **(V, E)**, **V** ένα μη κενό σύνολο
E διμελής σχέση στο **V**
- **Μη-κατευθυνόμενος γράφος :** σχέση **E** συμμετρική
- **V :** κορυφές (vertices) ή κόμβοι (nodes)
- **E :** ακμές (edges)

Εφαρμογές Γράφων & Δικτύων

Γράφος - Δίκτυο



Περίπατος του Ιππότη/Αλόγου
Διαδρομή Hamilton

GPS – Navigation



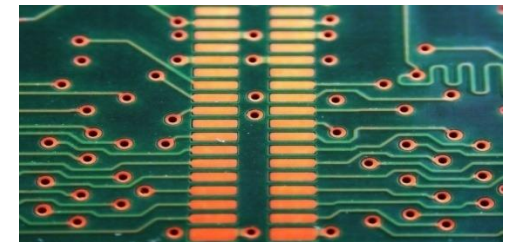
Χάρτες Χρωματισμός



Συνδεσιμότητα

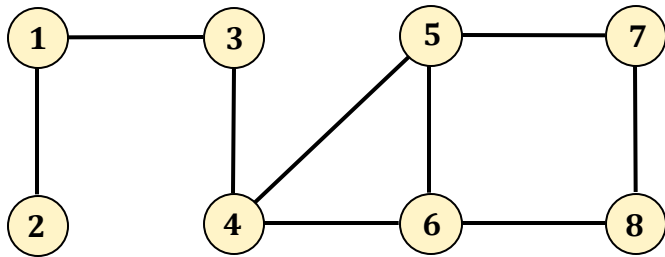


Συντομότερες διαδρομές

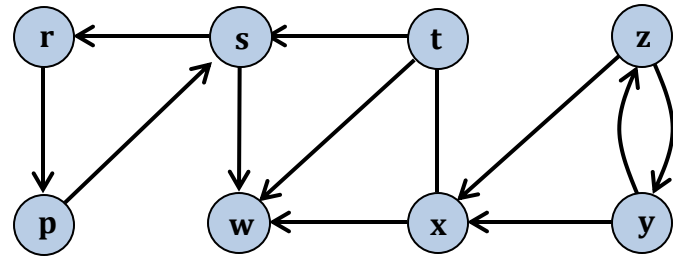


Επιπεδότητα

Γράφοι: ορολογία



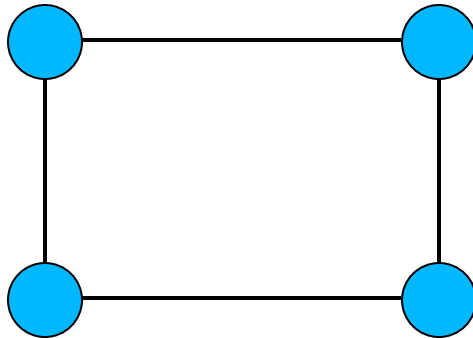
$$G_1 = (V_1, E_1)$$



$$G_2 = (V_2, E_2)$$

- **Γειτονικές (adjacent) κορυφές:** συνδέονται με ακμή, π.χ. 4 και 6
- **Άκρα (endpoints) ακμής**
- **Προσπίπτουσα (incident) ακμή σε κόμβο**
- **Γειτονικές ακμές:** προσπίτουν στον ίδιο κόμβο

Γράφοι: ορολογία

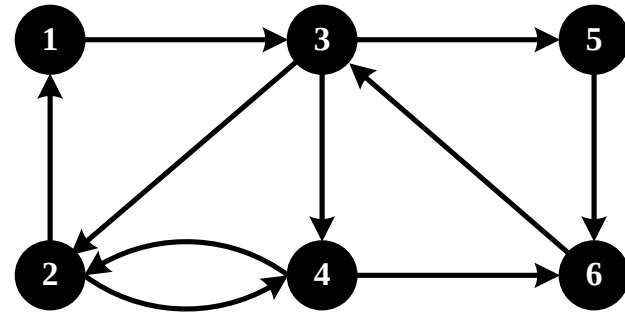
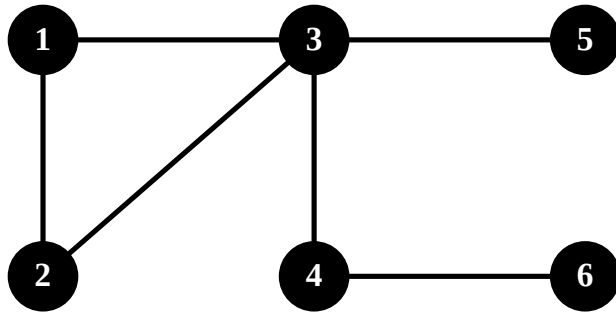


2-κανονικός γράφος

- **Βαθμός (degree, valence) κορυφής v :** ο αριθμός των ακμών που προσπίπτουν στην v , $\text{deg}(v)$
- Ένας (μη κατευθυνόμενος) γράφος όπου $\text{deg}(v)=k$ για κάθε κορυφή v , λέγεται **k -κανονικός (k -regular)**
- Σημαντική ιδιότητα: $\sum \text{deg}(v) = 2|E|$ (σε μη κατευθ/νους γράφους)
- Σε κατευθυνόμενο γράφο: $\text{in-deg}(v)$, $\text{out-deg}(v)$

$$\sum \text{out-deg}(v) = \sum \text{in-deg}(v) = |E|$$

Διαδρομές σε γράφους

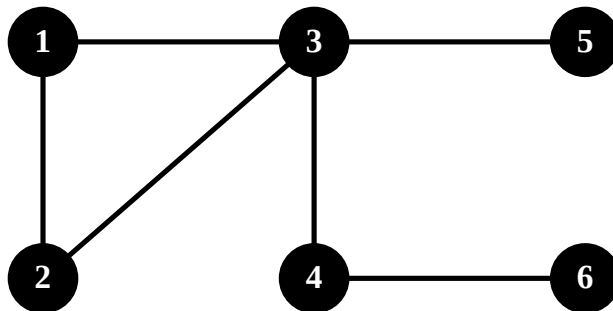


- ❑ **Δρόμος:** έγκυρη ακολουθία από κορυφές-ακμές
- ❑ **Μονοπάτι:** δρόμος χωρίς επαναλήψεις ακμών
Απλό μονοπάτι: μονοπάτι χωρίς επαναλήψεις κορυφών
- ❑ **Κύκλος:** κλειστό μονοπάτι
Απλός κύκλος: απλό κλειστό μονοπάτι
- ❑ **Μήκος δρόμου:** το πλήθος των ακμών του

Αναπαράσταση γράφων

□ ... 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-κατευθυνόμενος: **συμμετρικός** πίνακας
- Χώρος: $\Theta(n^2)$
- Προσπέλαση γειτόνων: $\Theta(n)$
- **Άμεσος** έλεγχος ύπαρξης ακμής: $O(1)$

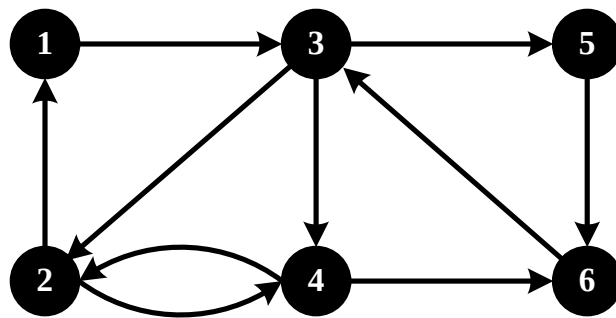


	1	2	3	4	5	6
1	0	1	1	0	0	0
2	1	0	1	0	0	0
3	1	1	0	1	1	0
4	0	0	1	0	0	1
5	0	0	1	0	0	0
6	0	0	0	1	0	0

Αναπαράσταση γράφων

□ ... με **πίνακα γειτνίασης**:
$$A[i, j] = \begin{cases} 1 & (v_i, v_j) \in E \\ 0 & (v_i, v_j) \notin E \end{cases}$$

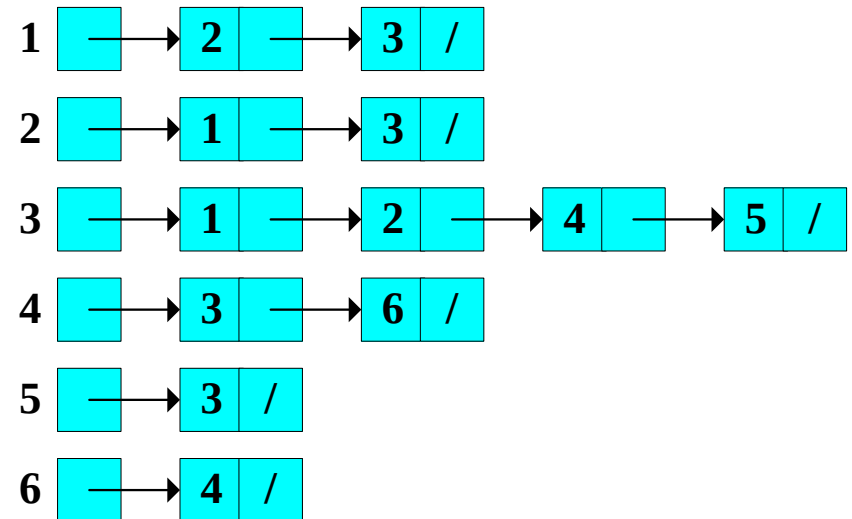
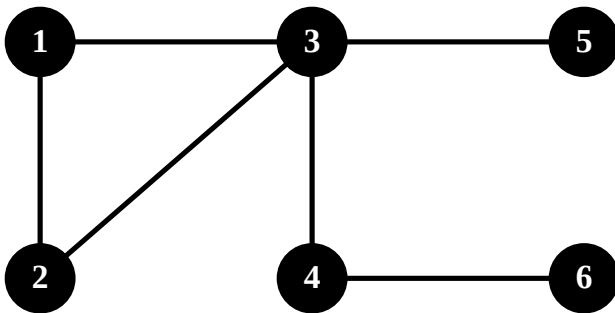
- Αν έχουμε βάρη, $A[i, j] = w(v_i, v_j)$
- Μη-κατευθυνόμενος: **μη-συμμετρικός** πίνακας
- Χώρος: $\Theta(n^2)$
- Προσπέλαση γειτόνων: $\Theta(n)$
- **Άμεσος** έλεγχος ύπαρξης ακμής: $O(1)$



	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

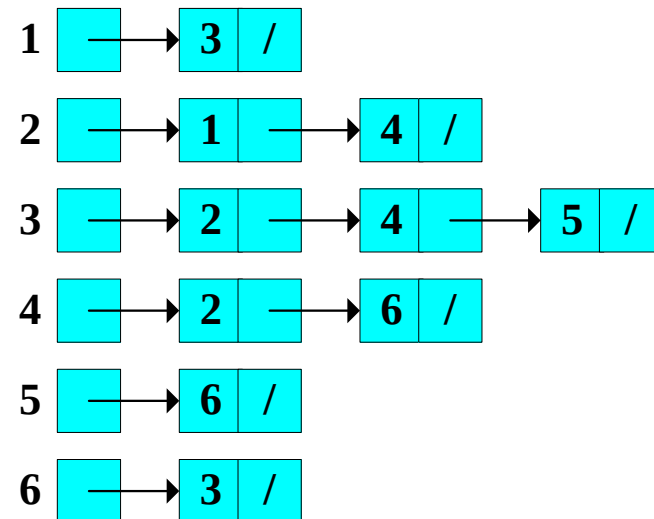
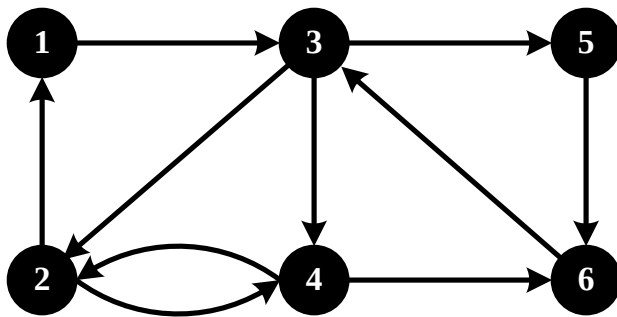
Αναπαράσταση γράφων

- ... με **λίστες γειτνίασης**: γειτονικές κορυφές σε λίστες
 - Αν έχουμε βάρη, τα αποθηκεύουμε στους κόμβους
 - Χώρος: $\Theta(m)$
 - Προσπέλαση γειτόνων: $\Theta(\deg(u))$
 - Έλεγχος ύπαρξης ακμής: $O(\deg(u))$



Αναπαράσταση γράφων

- ... με **λίστες γειτνίασης**: γειτονικές κορυφές σε λίστες
 - Αν έχουμε βάρη, τα αποθηκεύουμε στους κόμβους
 - Χώρος: $\Theta(m)$
 - Προσπέλαση γειτόνων: $\Theta(\deg(u))$
 - Έλεγχος ύπαρξης ακμής: $O(\deg(u))$



Γράφοι: συνεκτικότητα

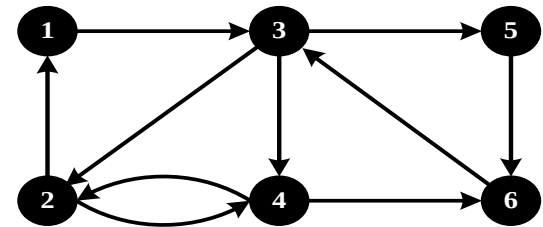
- Ένας μη κατευθυνόμενος γράφος λέγεται **συνεκτικός (connected)** αν υπάρχει δρόμος μεταξύ οποιωνδήποτε δύο κορυφών του

Σε συνεκτικό γράφο ισχύει: $n - 1 \leq e \leq \frac{n(n-1)}{2}, n = |V|, e = |E|.$

- Ένας κατευθυνόμενος γράφος λέγεται

- **ισχυρά συνεκτικός (strongly connected)**

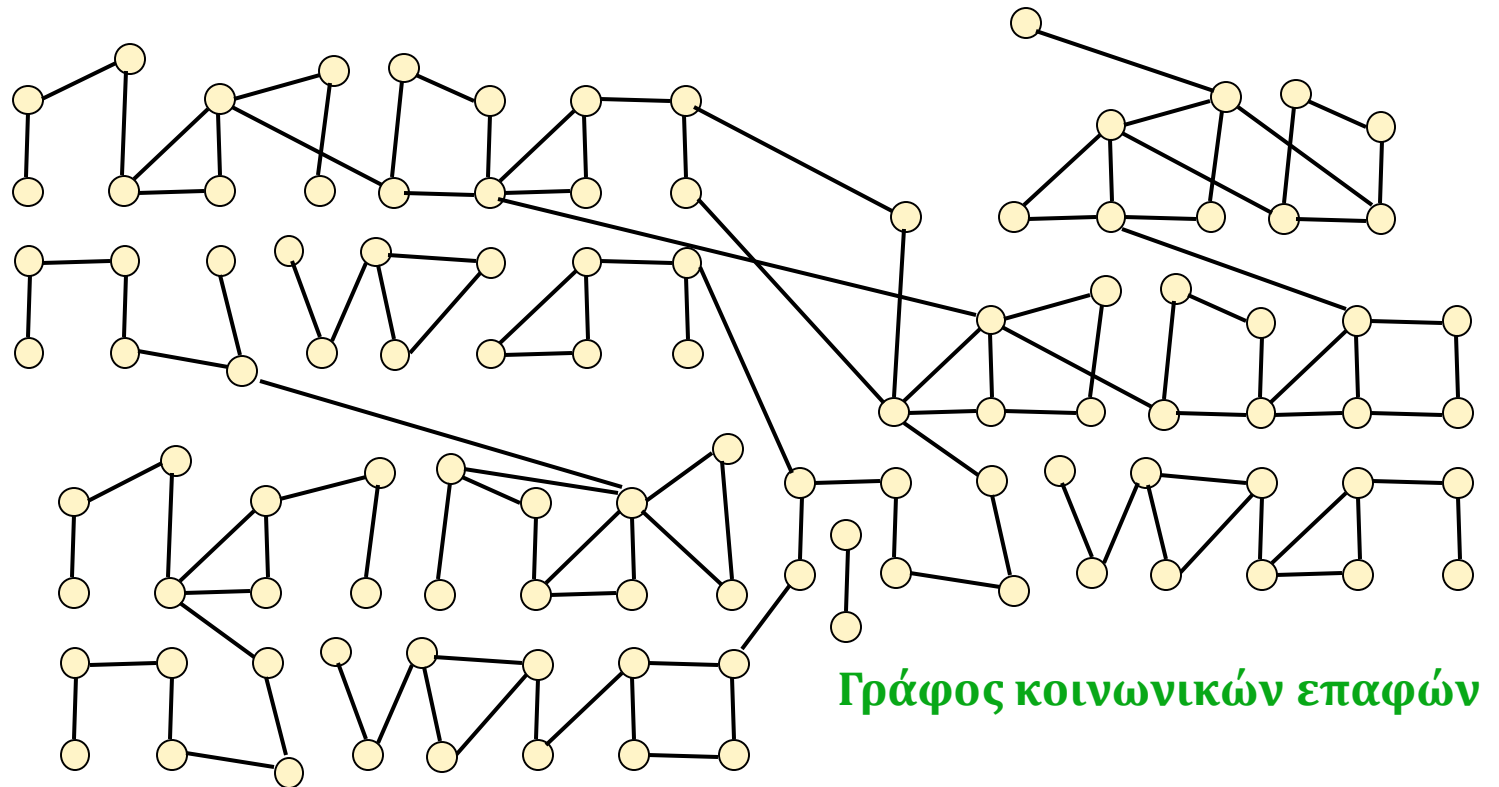
αν υπάρχει δρόμος μεταξύ οποιωνδήποτε δύο κορυφών του ακολουθώντας τις κατευθύνσεις των ακμών



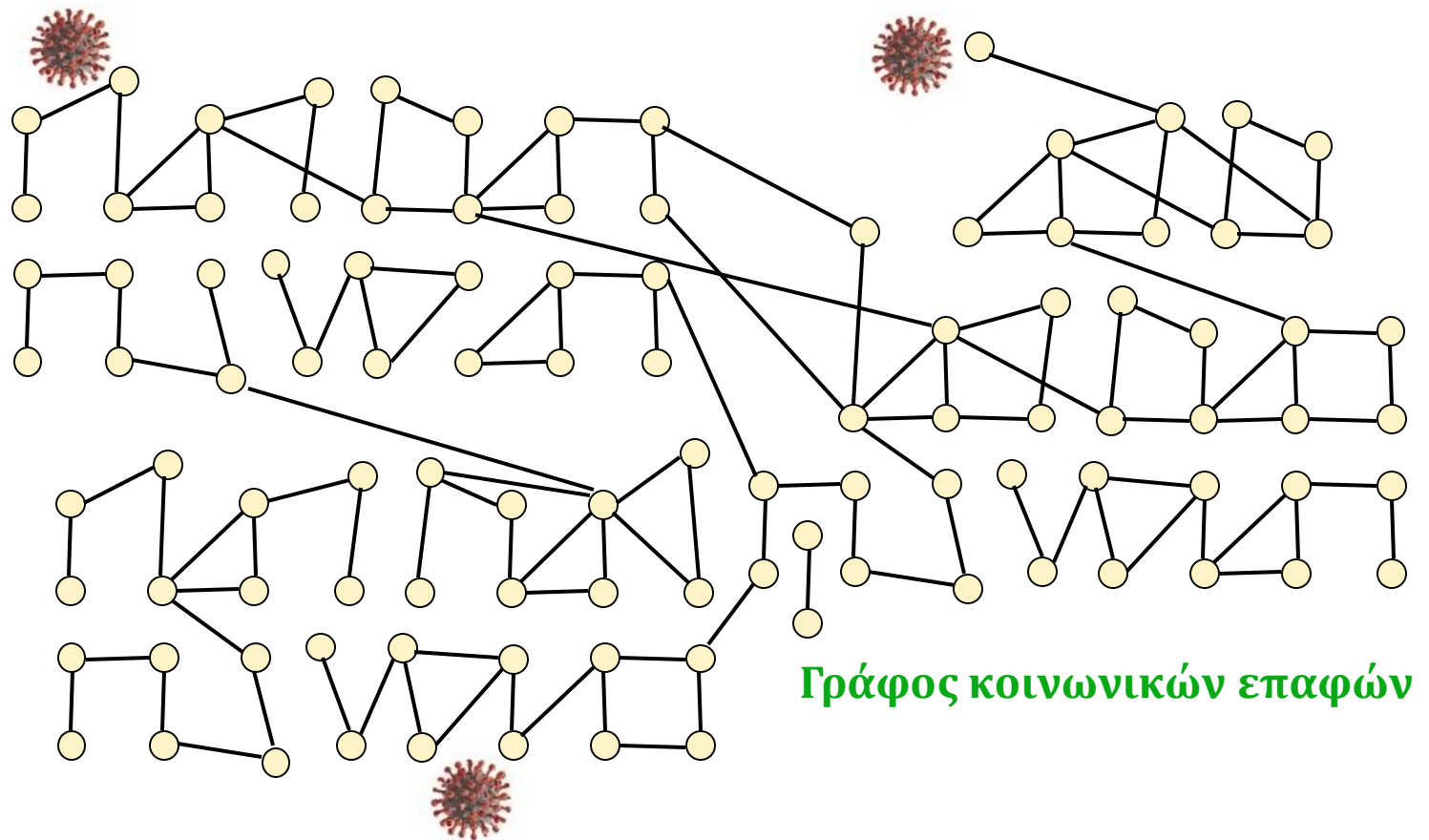
- **ασθενώς συνεκτικός (weakly connected)**

αν υπάρχει δρόμος μεταξύ οποιωνδήποτε δύο κορυφών του αγνοώντας τις κατευθύνσεις των ακμών

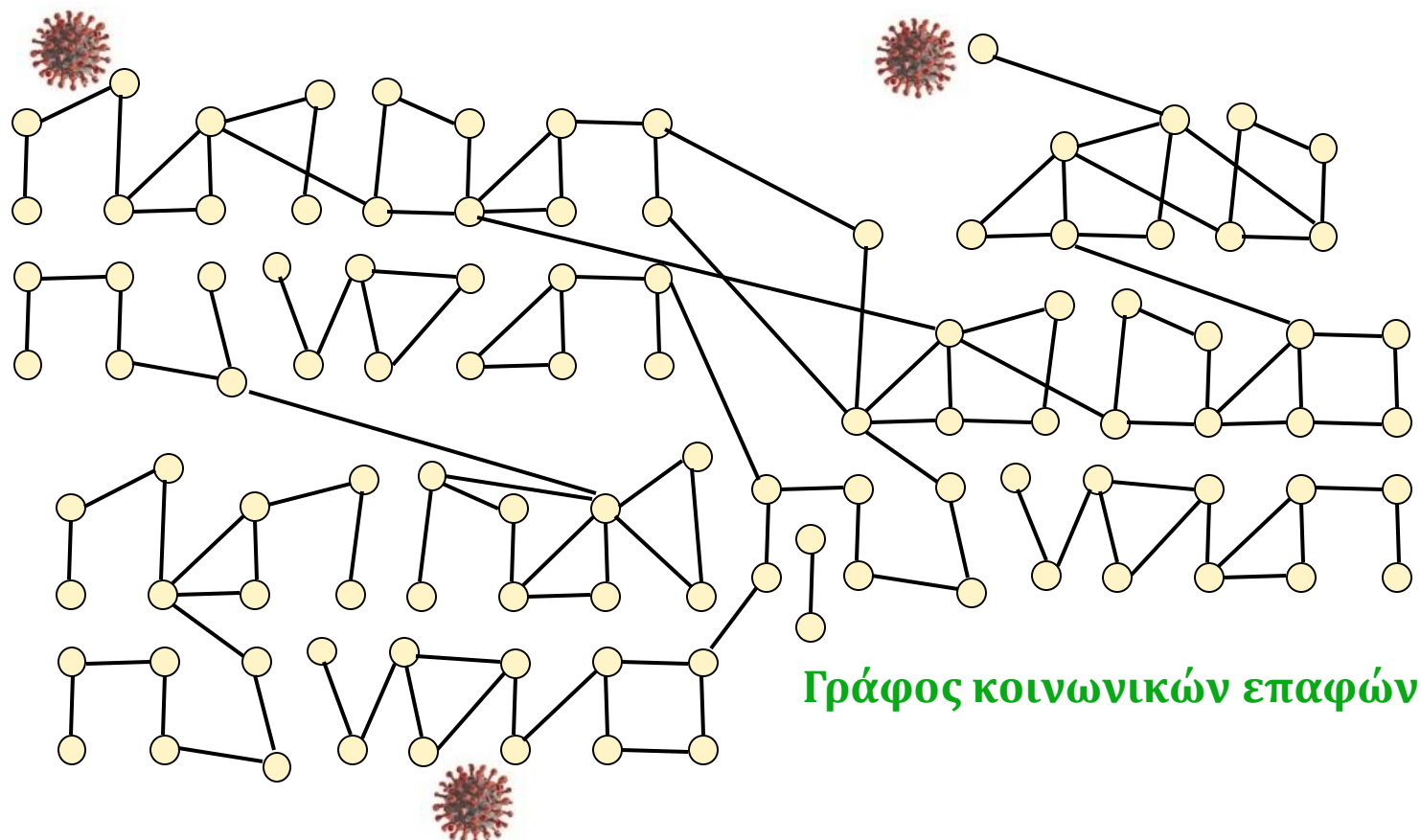
Εξερεύνηση Γραφημάτων



Εξερεύνηση Γραφημάτων

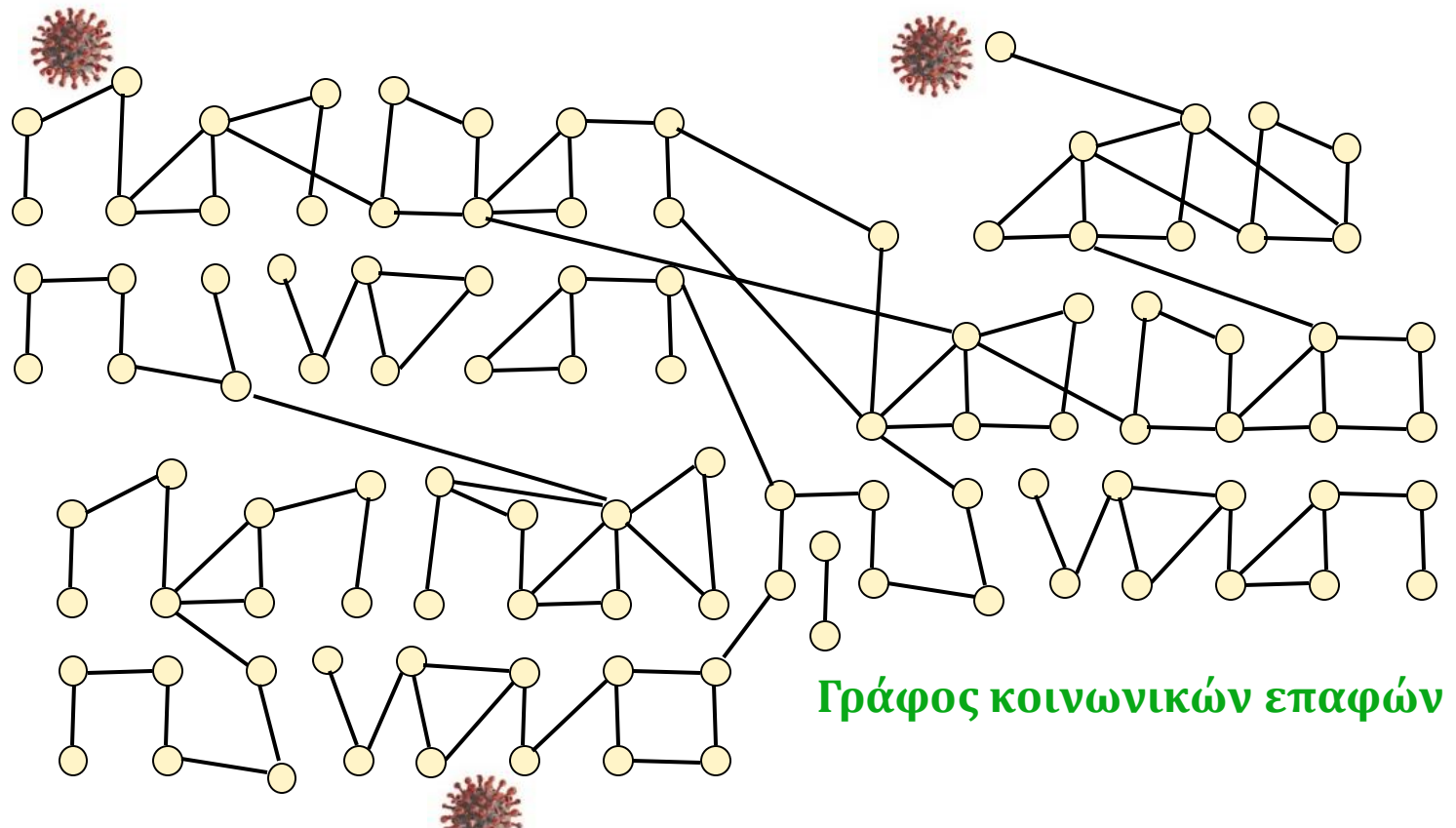


Εξερεύνηση Γραφημάτων



Ποιοί μπορεί να κολλήσουν **covid-19**; Πόσο γρήγορα;
Ποιούς πρέπει να ειδοποιήσουμε πρώτα;

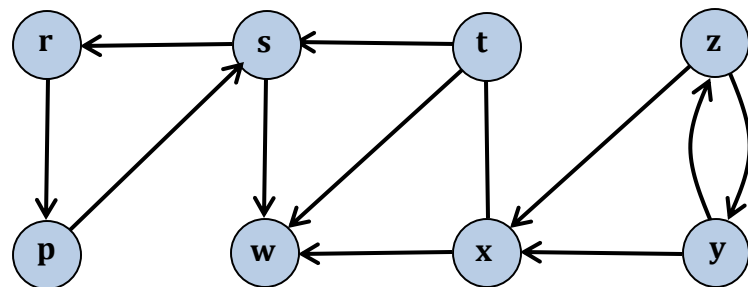
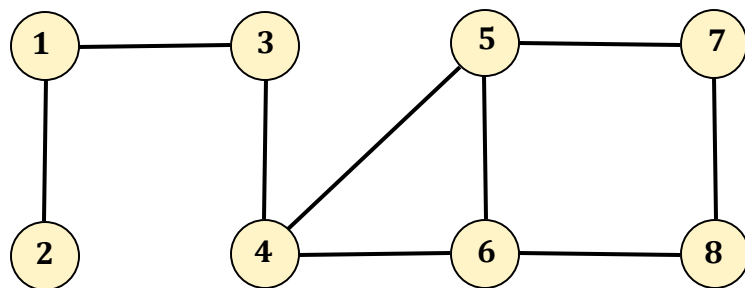
Εξερεύνηση Γραφημάτων



Κύριο ερώτημα: δεδομένου γράφου και κόμβου s , ποιοί κόμβοι του γράφου είναι προσβάσιμοι από τον s και πώς;

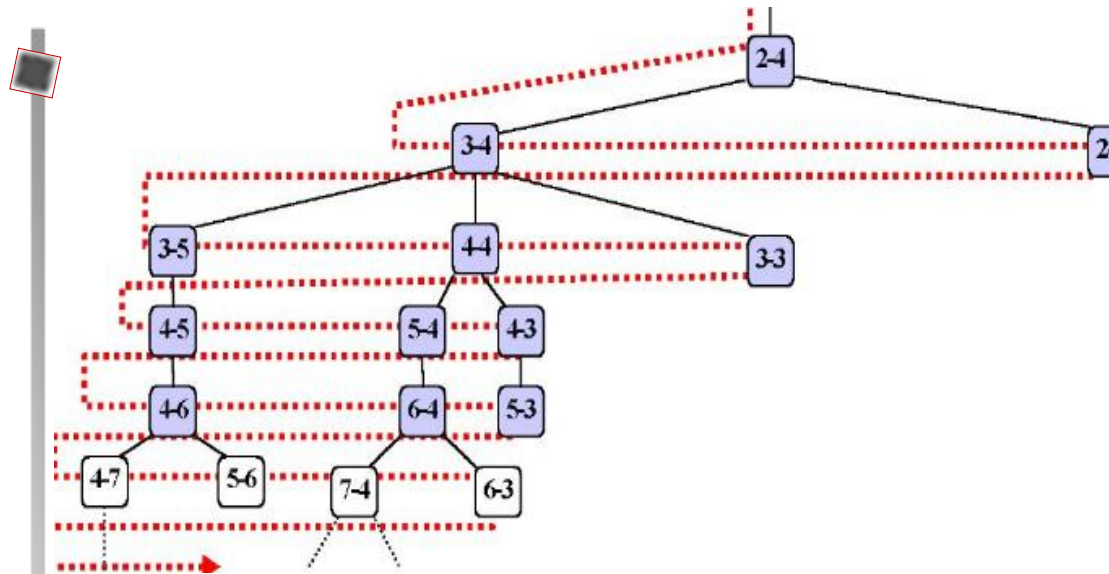
Γνωστό και ως πρόβλημα **προσβασιμότητας/διάσχισης/αναζήτησης**

Εξερεύνηση Γραφημάτων



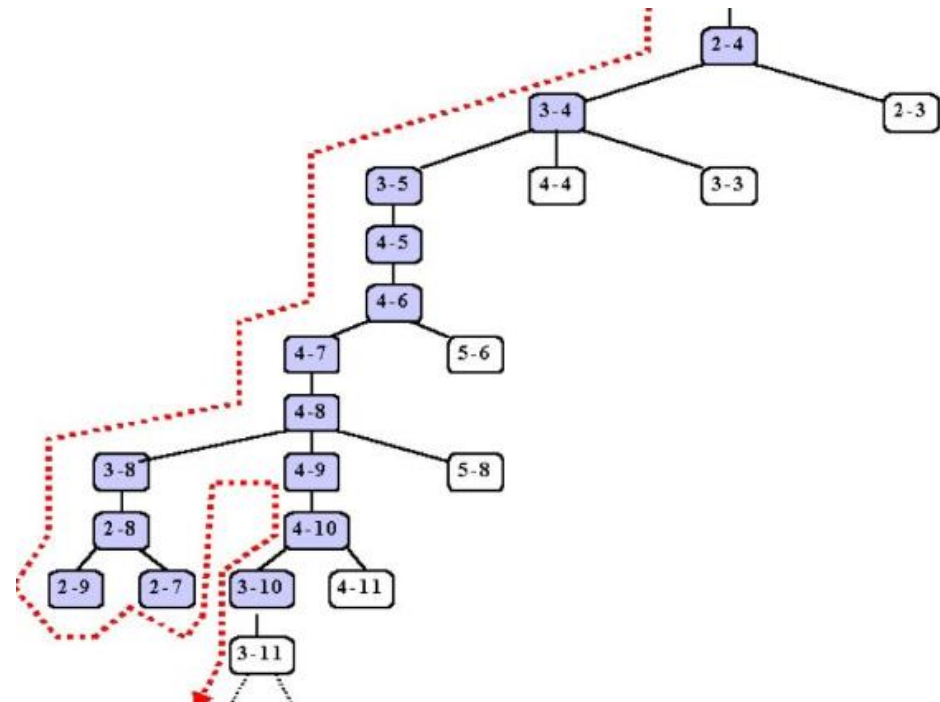
- Ορίζεται και σε μη κατευθυνόμενα (undirected) και σε κατευθυνόμενα (directed) γραφήματα
- Δύο βασικές μέθοδοι:
 - Εξερεύνηση κατά Πλάτος -- Breadth-First Search (BFS)
 - Εξερεύνηση κατα Βάθος -- Depth-First Search (DFS)

Εξερεύνηση Γραφημάτων

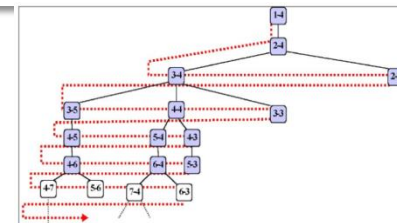


Εξερεύνηση BFS

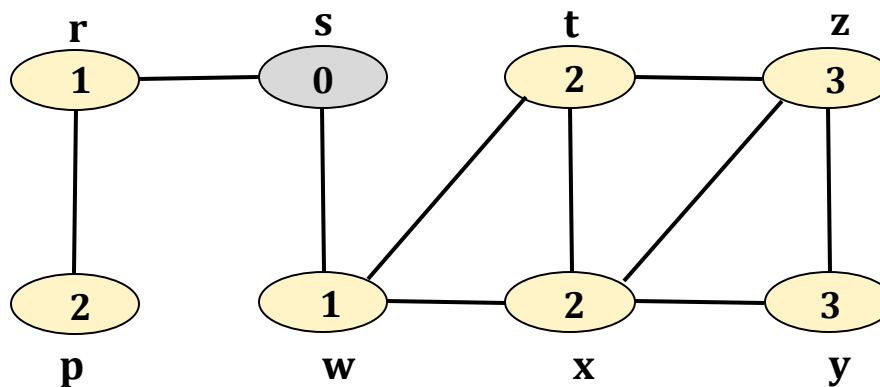
Εξερεύνηση DFS



Εξερεύνηση κατά Πλάτος - BFS



- Έστω $G = (V, E)$ ένας γράφος και $s \in V$ ένας κόμβος εκκίνησης

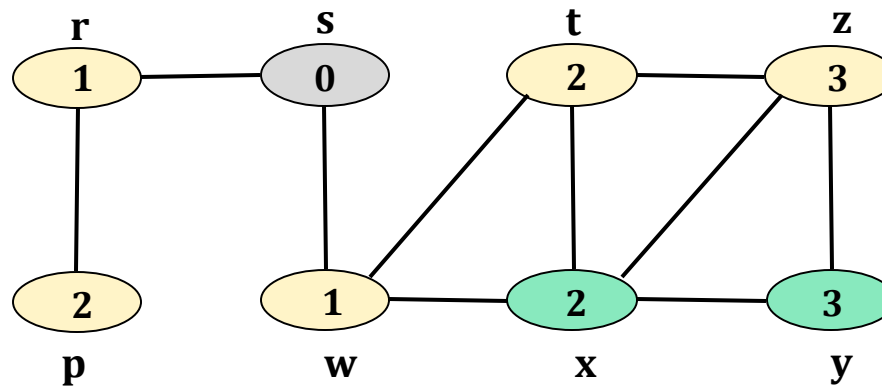


- $\mathbf{d}_s(\mathbf{x})$: ελάχιστη απόσταση του κόμβου $\mathbf{x} \in V$ από τον s

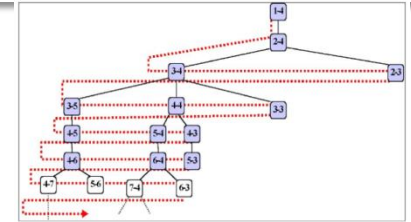
Εξερεύνηση κατά Πλάτος - BFS

Βασική Ιδέα BFS

- Ο κόμβος **x** εξερευνάται **πριν** τον **y** σε μια BFS, εάν $d_s(x) < d_s(y)$, δηλαδή η BFS εξερευνά **πρώτα** τους πλησιέστερους κόμβους

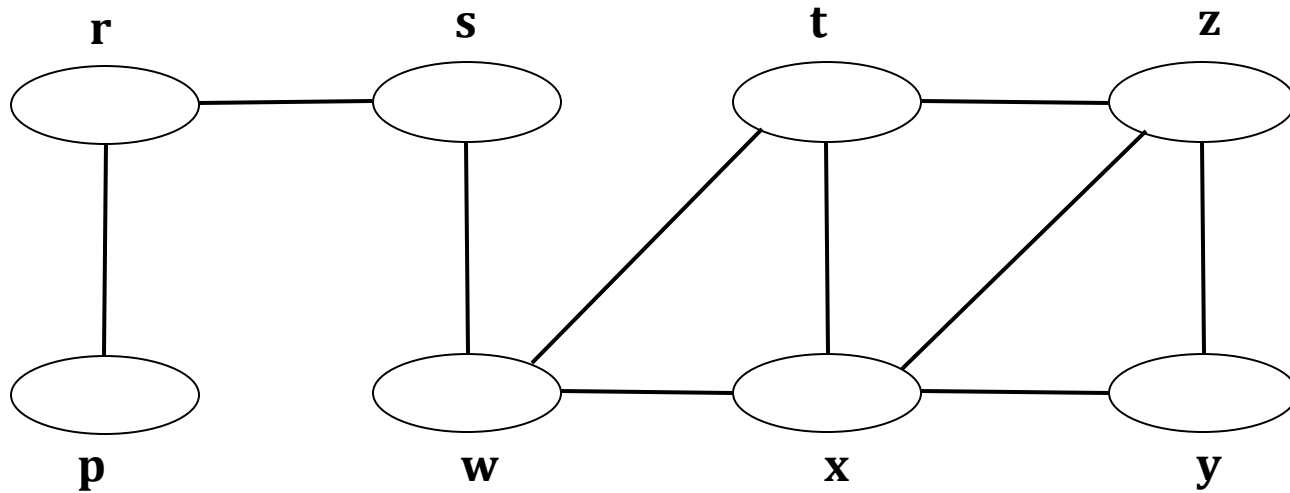


- Ο αλγόριθμος **BFS** εξερευνά κάθε κόμβο **x** του G για τον οποίο υπάρχει μονοπάτι: $s \rightarrow \dots \rightarrow x$



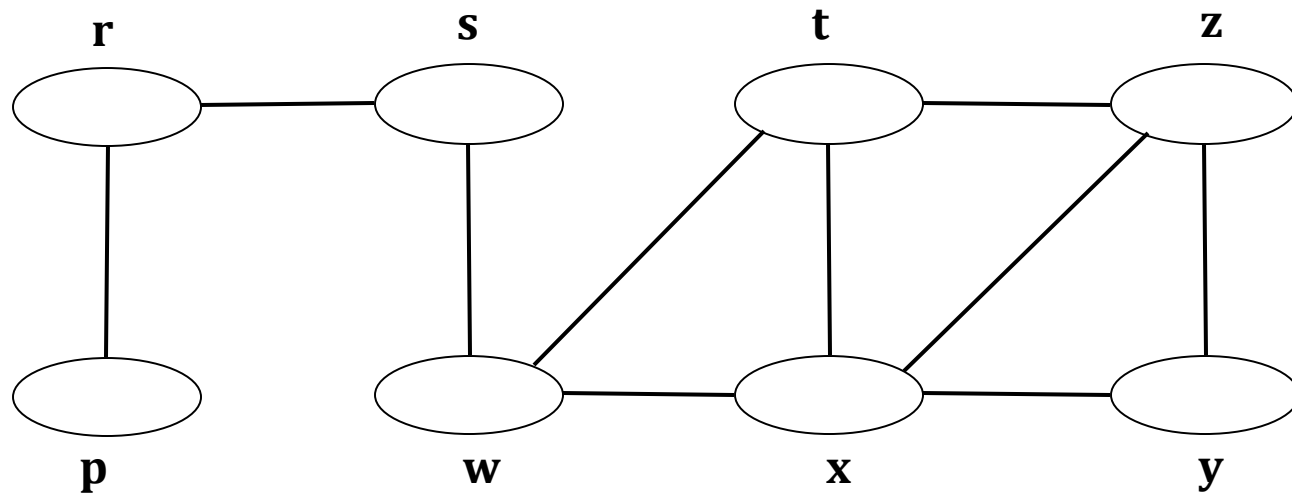
Εξερεύνηση κατά Πλάτος - BFS

Παράδειγμα



Εξερεύνηση κατά Πλάτος - BFS

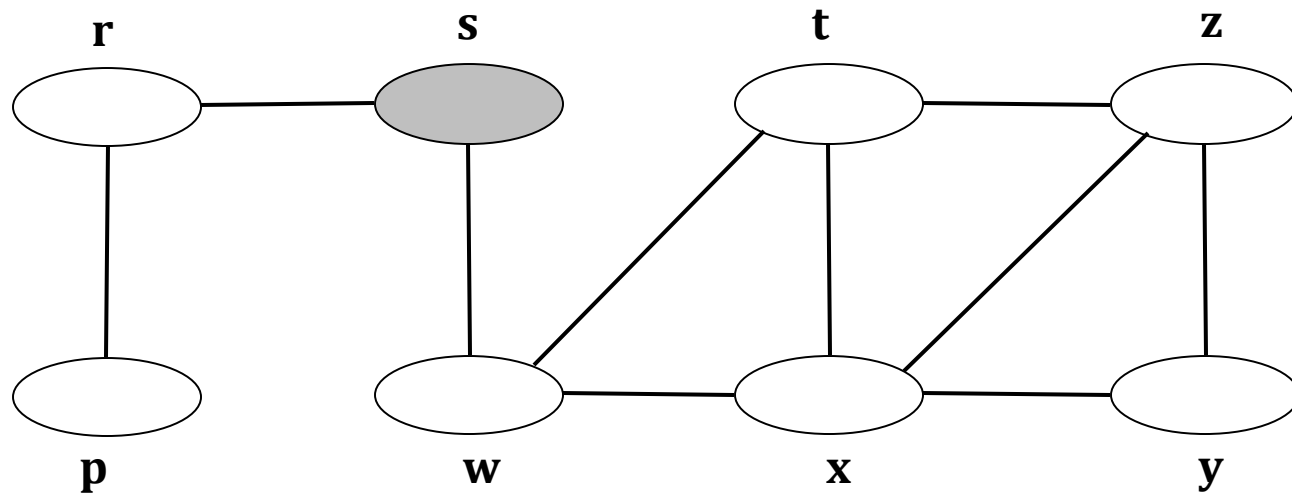
Παράδειγμα



- Σημαντικά συστατικά: χρήση **ουράς** (FIFO) για αποδοτικότητα, χρήση **δείκτη** σε προηγούμενο κόμβο.

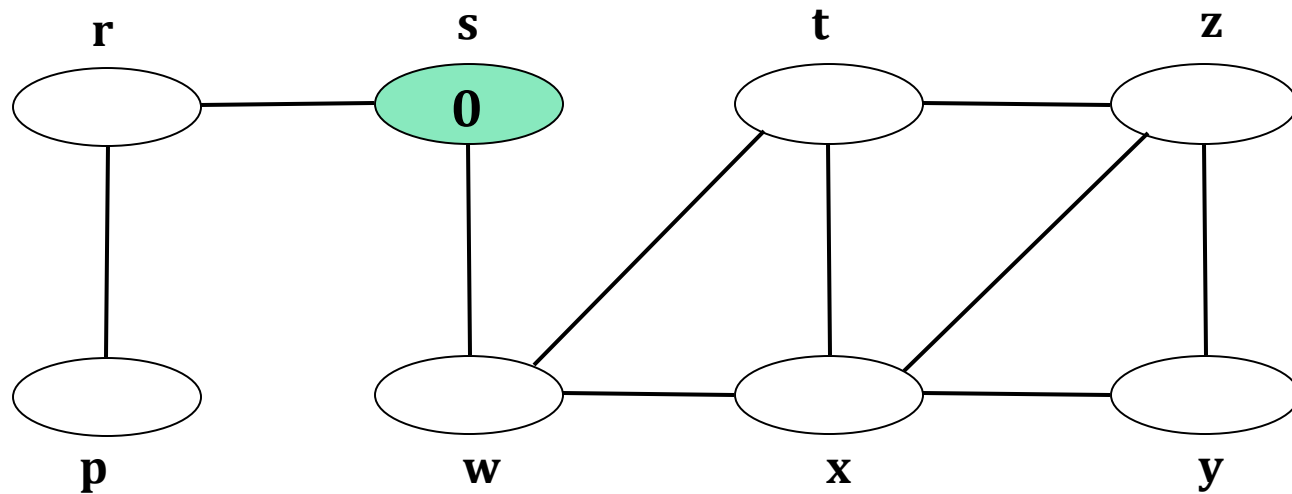
Εξερεύνηση κατά Πλάτος - BFS

Παράδειγμα



Εξερεύνηση κατά Πλάτος - BFS

Παράδειγμα

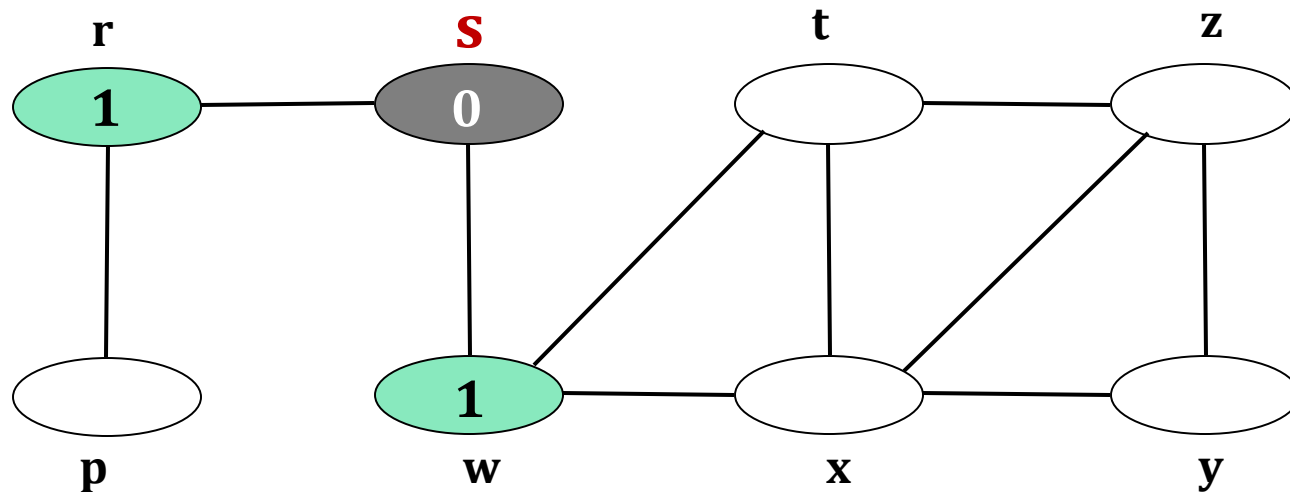


$\pi(s) = \text{nil}$

Q : **s**

Εξερεύνηση κατά Πλάτος - BFS

Παράδειγμα



$\pi(s) = \text{nil}$

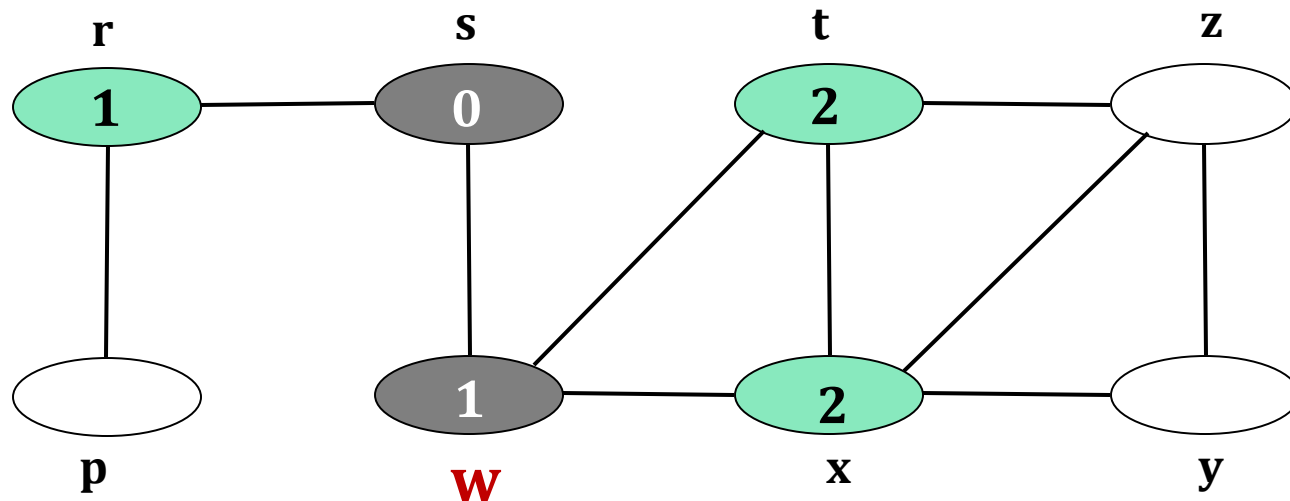
$u \leftarrow \text{head}(Q) = \mathbf{s} :$

s w r

$\pi(\mathbf{w}) = \mathbf{s} \quad \pi(\mathbf{r}) = \mathbf{s}$

Εξερεύνηση κατά Πλάτος - BFS

Παράδειγμα



$\pi(s) = \text{nil}$

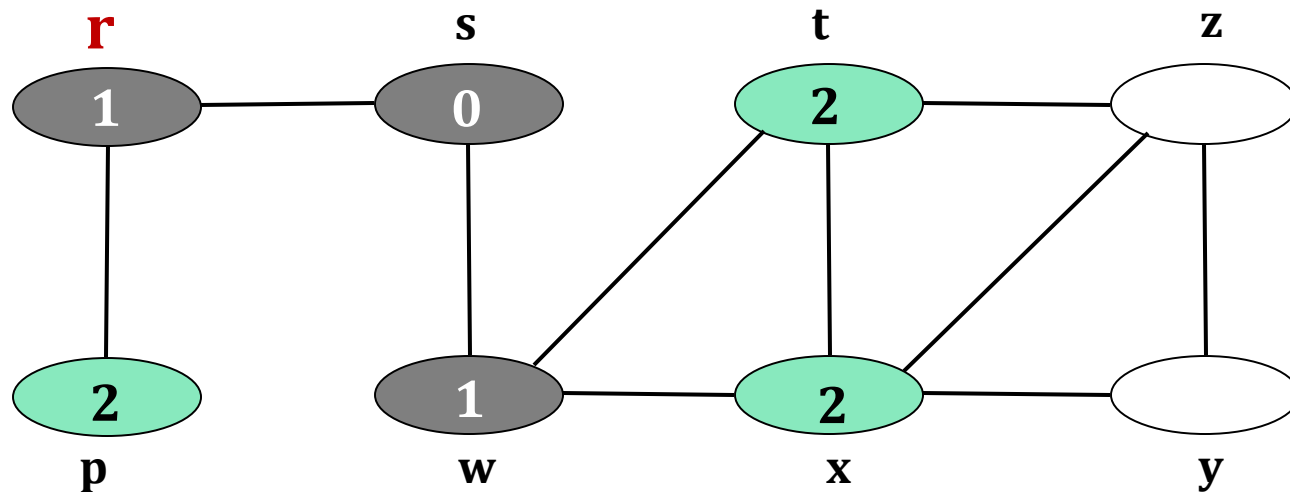
$u \leftarrow \text{head}(Q) = w :$

w r t x

$\pi(w) = s \quad \pi(r) = s \quad \pi(t) = w \quad \pi(x) = w$

Εξερεύνηση κατά Πλάτος - BFS

Παράδειγμα



$\pi(s) = \text{nil}$

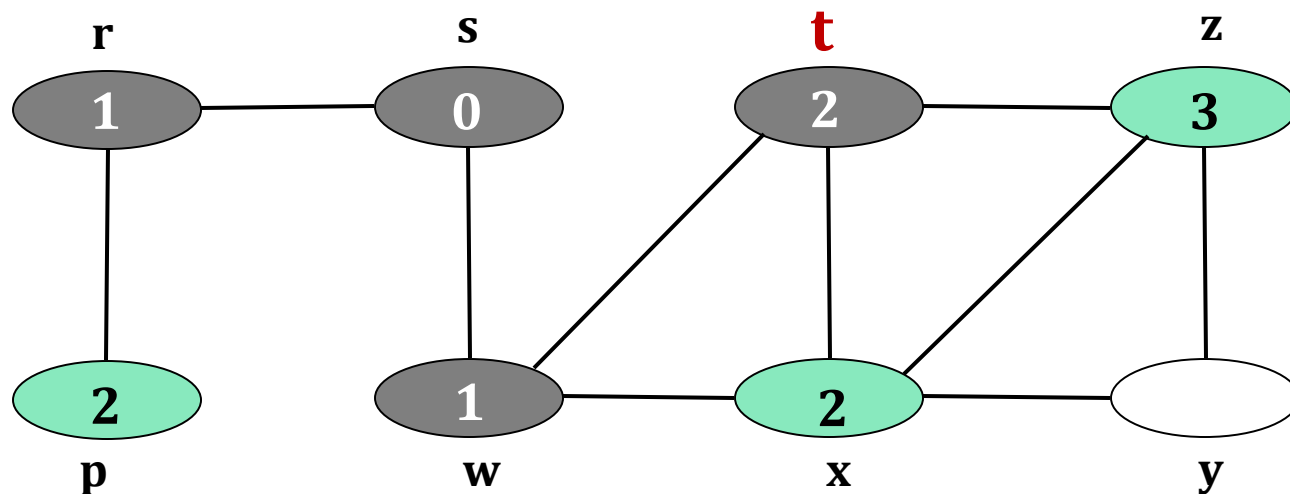
$u \leftarrow \text{head}(Q) = \mathbf{r} :$

r t x **p**

$\pi(w) = s \quad \pi(r) = s \quad \pi(t) = w \quad \pi(x) = w \quad \pi(\mathbf{p}) = \mathbf{r}$

Εξερεύνηση κατά Πλάτος - BFS

Παράδειγμα



$\pi(s) = \text{nil}$

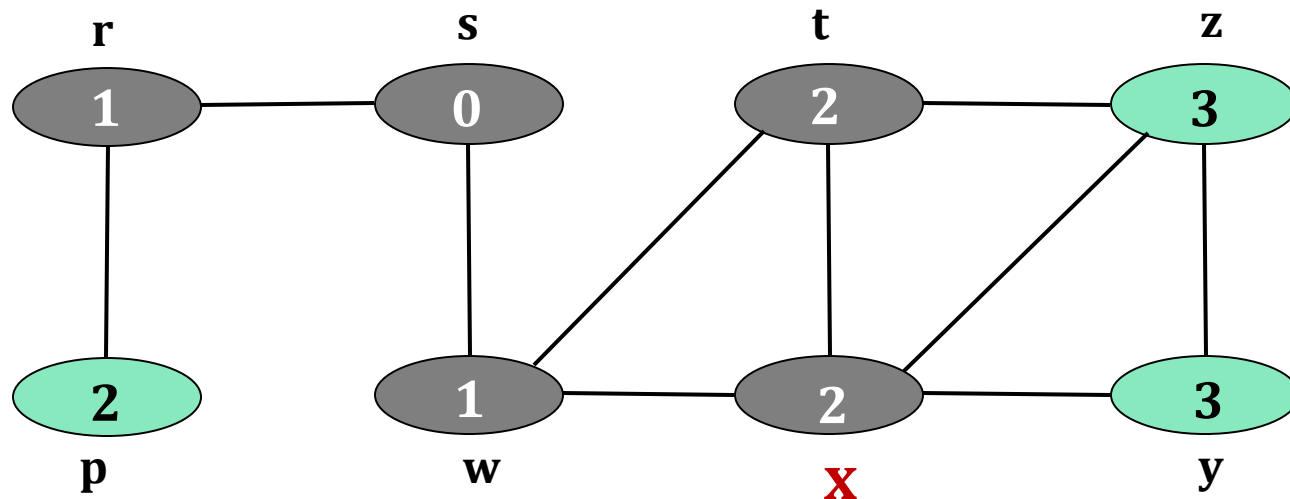
$u \leftarrow \text{head}(Q) = \mathbf{t}$:

t x p **z**

$\pi(w) = s$ $\pi(r) = s$ $\pi(\mathbf{t}) = w$ $\pi(x) = w$ $\pi(p) = r$ $\pi(\mathbf{z}) = \mathbf{t}$

Εξερεύνηση κατά Πλάτος - BFS

Παράδειγμα



$\pi(s) = \text{nil}$

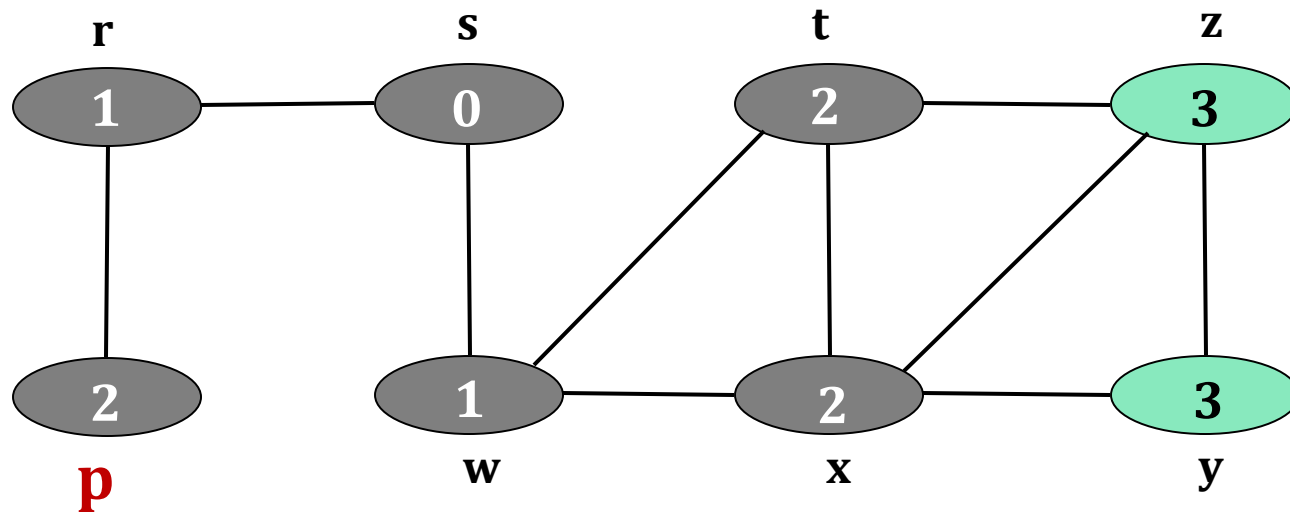
$u \leftarrow \text{head}(Q) = \mathbf{x} :$

x **p** **z** **y**

$\pi(w) = s$ $\pi(r) = s$ $\pi(t) = w$ $\pi(x) = w$ $\pi(p) = r$ $\pi(z) = t$ $\pi(\mathbf{y}) = \mathbf{x}$

Εξερεύνηση κατά Πλάτος - BFS

Παράδειγμα



$\pi(s) = \text{nil}$

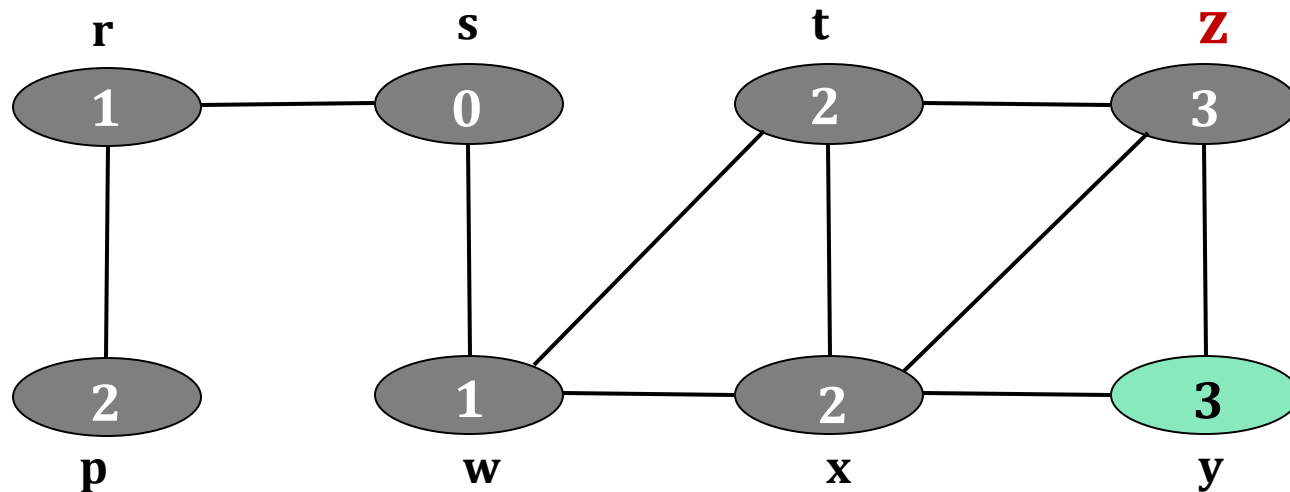
$u \leftarrow \text{head}(Q) = \mathbf{p} :$

p z y

$\pi(w) = s \quad \pi(r) = s \quad \pi(t) = w \quad \pi(x) = w \quad \pi(p) = r \quad \pi(z) = t \quad \pi(y) = x$

Εξερεύνηση κατά Πλάτος - BFS

Παράδειγμα



$\pi(s) = \text{nil}$

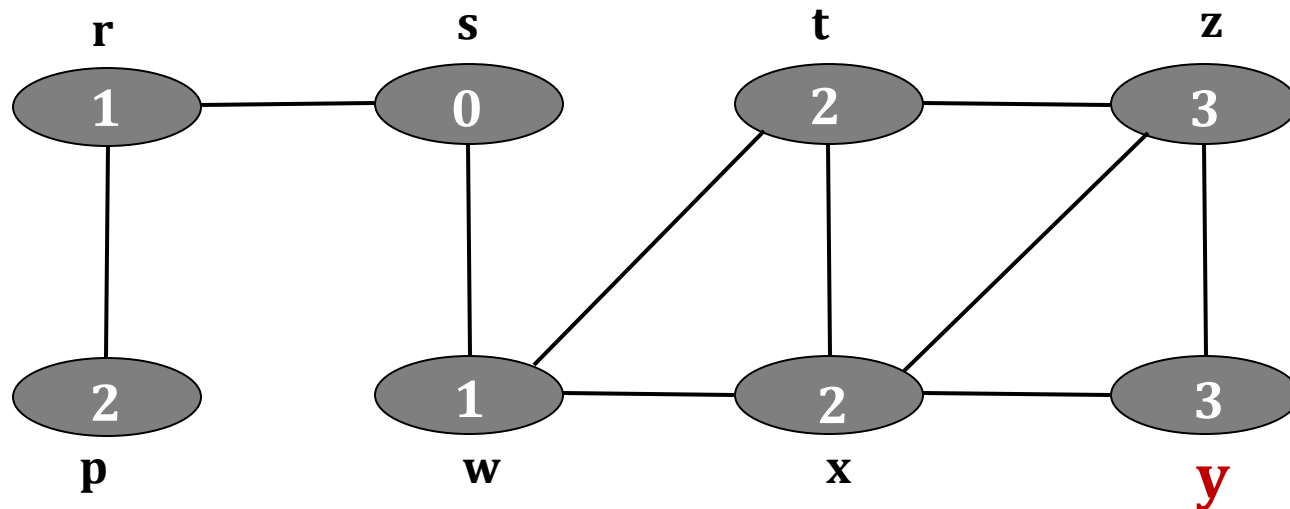
$u \leftarrow \text{head}(Q) = \mathbf{z} :$

z y

$\pi(w) = s \quad \pi(r) = s \quad \pi(t) = w \quad \pi(x) = w \quad \pi(p) = r \quad \pi(z) = t \quad \pi(y) = x$

Εξερεύνηση κατά Πλάτος - BFS

Παράδειγμα



$\pi(s) = \text{nil}$

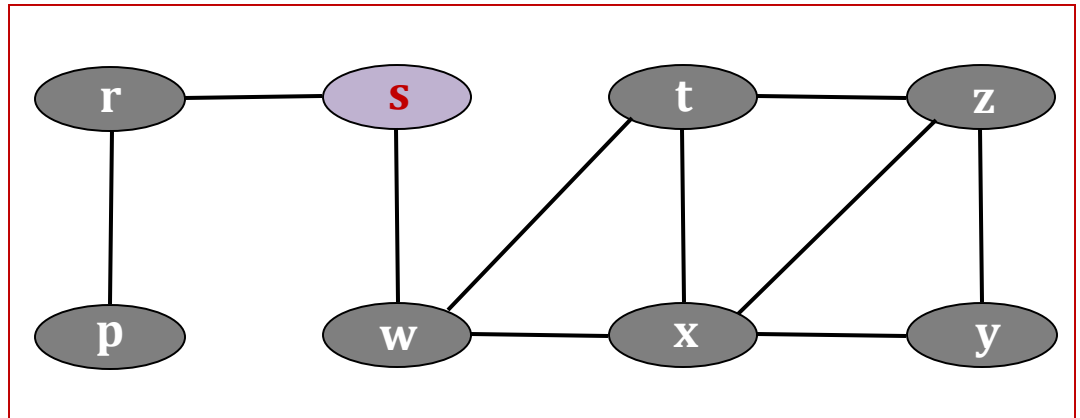
$u \leftarrow \text{head}(Q) = \mathbf{y} :$

y

$\pi(w) = s \quad \pi(r) = s \quad \pi(t) = w \quad \pi(x) = w \quad \pi(p) = r \quad \pi(z) = t \quad \pi(y) = x$

Εξερεύνηση κατά Πλάτος - BFS

Παράδειγμα



BFS = (s, w, r, t, x, p, z, y)

$\pi(s) = \text{nil}$

$u \leftarrow \text{head}(Q) = \text{nil} :$

$\pi(w) = s \quad \pi(r) = s \quad \pi(t) = w \quad \pi(x) = w \quad \pi(p) = r \quad \pi(z) = t \quad \pi(y) = x$

Εξερεύνηση κατά Πλάτος (BFS): αλγόριθμος



BFS($G=(V,E)$)

1. **for each** vertex $v \in V$ **do**
2. $\text{status}[v] \leftarrow \text{UNVISITED}; \pi(v) \leftarrow \text{nil}$
3. **while** $\exists s$ with $\text{status}[s] = \text{UNVISITED}$ **do**
4. BFS-Help(s)

BFS-Help(r)

$\text{status}[r] \leftarrow \text{VISITED}; \text{push}(Q, r)$ // Q : ουρά (FIFO)

while nonempty Q

$u \leftarrow \text{head}(Q); \text{pop}(Q);$

for each vertex $x \in \text{adj}(u)$ **do**

if $\text{status}[x] = \text{UNVISITED}$

$\text{status}[x] \leftarrow \text{VISITED}; \pi(x) \leftarrow u$

$\text{push}(Q, x)$

$\text{status}[u] \leftarrow \text{EXPLORED};$ // όχι πάντα απαραίτητη

Εξερεύνηση κατά Πλάτος - BFS

● Πολυπλοκότητα BFS

- ✓ Κάθε κόμβος $v \in V$ εισέρχεται στην ουρά Q ακριβώς 1 φορά και διαγράφεται από αυτή ακριβώς 1 φορά.
- ✓ Κάθε **Εισαγωγή** / **Διαγραφή** απαιτεί **χρόνο $O(1)$** .
- ✓ Συνολικός χρόνος για όλες τις λειτουργίες της Q : **$O(|V|) = O(n)$**
- ✓ Η λίστα γειτνίασης $\text{adj}(v)$ κάθε κόμβου $v \in V$ εξερευνάται 1 μόνο φορά όταν ο κόμβος v εξάγεται από Q , κόστος **$O(\deg(v))$** .

$$\sum_{u \in V} \deg(u) = O(|E|)$$

$$O(|E|) = O(m)$$

χρόνος για
εξερεύνηση
όλων των λιστών.

- ✓ Άρα, πολυπλοκότητα BFS :

$$O(|V| + |E|) = O(n + m)$$

Εξερεύνηση κατά Πλάτος (BFS): υλοποίηση

Βάση: Αναπαράσταση γράφου με λίστα γειτνίασης

```
class Graph {  
public:  
    Graph(int n) ;  
    void addEdge(int u, int v) ;  
    bool hasEdge(int u, int v) const;  
    int vertices() const;  
    const list<int> & edges(int u) const;  
};
```

Προσοχή: υπάρχουν και άλλοι τρόποι υλοποίησης, επιλέξτε αυτόν που σας είναι πιο κατανοητός

Εξερεύνηση κατά Πλάτος (BFS): υλοποίηση

```
enum state { UNVISITED, VISITED, EXPLORED };  
void bfs(const Graph &g, vector<int> &p) {  
    int N = g.vertices();  
    vector<state> status(N);  
    for (int u = 0; u < N; ++u) {  
        p[u] = -1; status[u] = UNVISITED;  
    }  
    for (int u = 0; u < N; ++u)  
        if (status[u] == UNVISITED)  
            bfs_help(u, g, p, status);  
}
```

Προσοχή: υπάρχουν και άλλοι τρόποι υλοποίησης, επιλέξτε αυτόν που σας είναι πιο κατανοητός

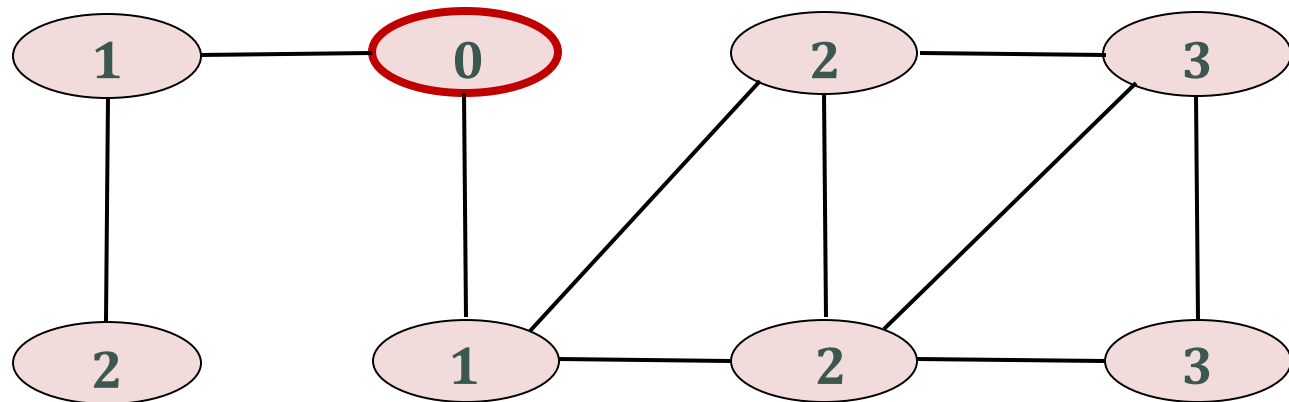
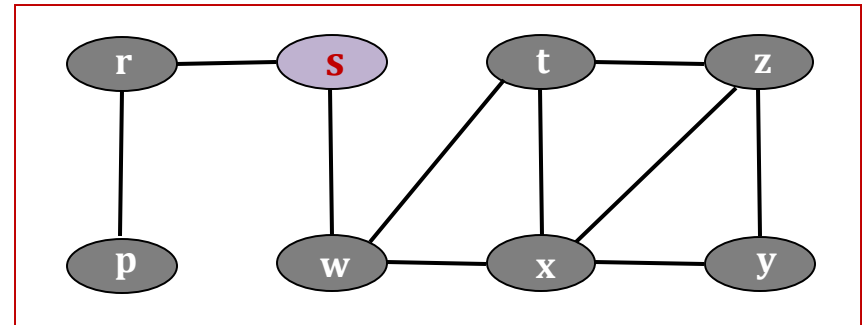
Εξερεύνηση κατά Πλάτος (BFS): υλοποίηση

```
void bfs_help(int u, const Graph &g,
              vector<int> &p, vector<state> &status) {
    queue<int> Q;
    status[u] = VISITED; Q.push(u);
    while (!Q.empty()) {
        int u = Q.front();
        Q.pop();
        for (int v: g.edges(u))
            if (status[v] == UNVISITED) {
                status[v] = VISITED; p[v] = u; Q.push(v);
            }
        status[u] = EXPLORED;
    }
}
```

Προσοχή: υπάρχουν και άλλοι τρόποι υλοποίησης, επιλέξτε αυτόν που σας είναι πιο κατανοητός

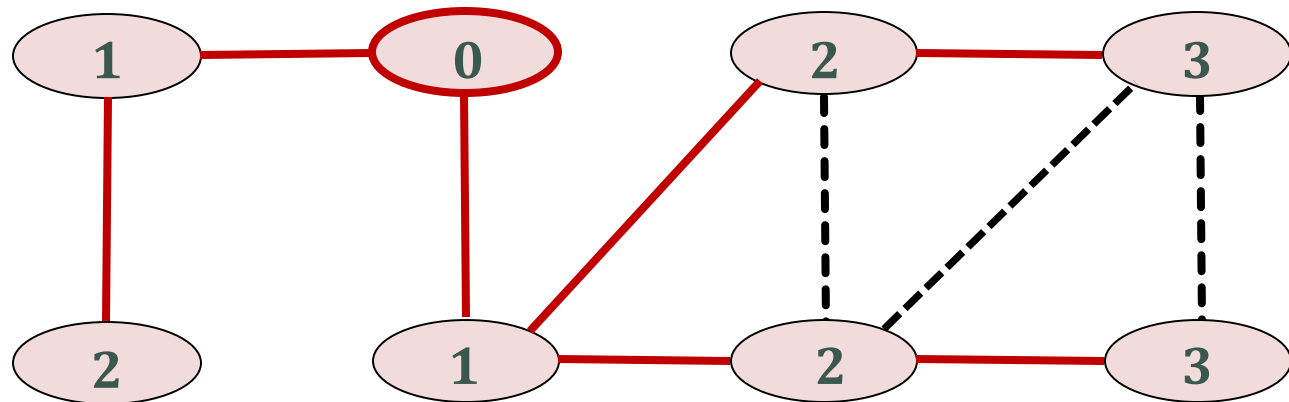
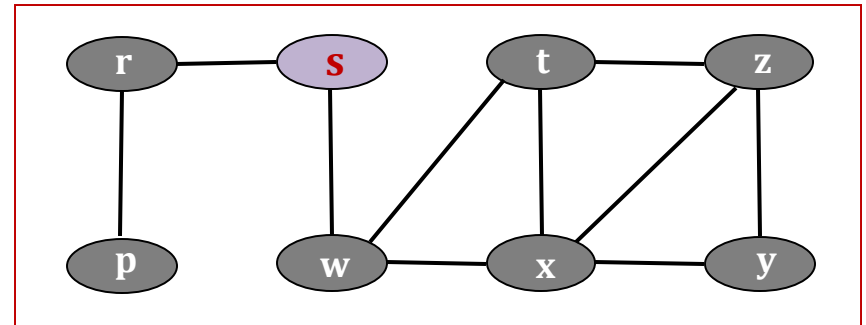
Εξερεύνηση κατά Πλάτος - BFS

BFS-δένδρο (δάσος)



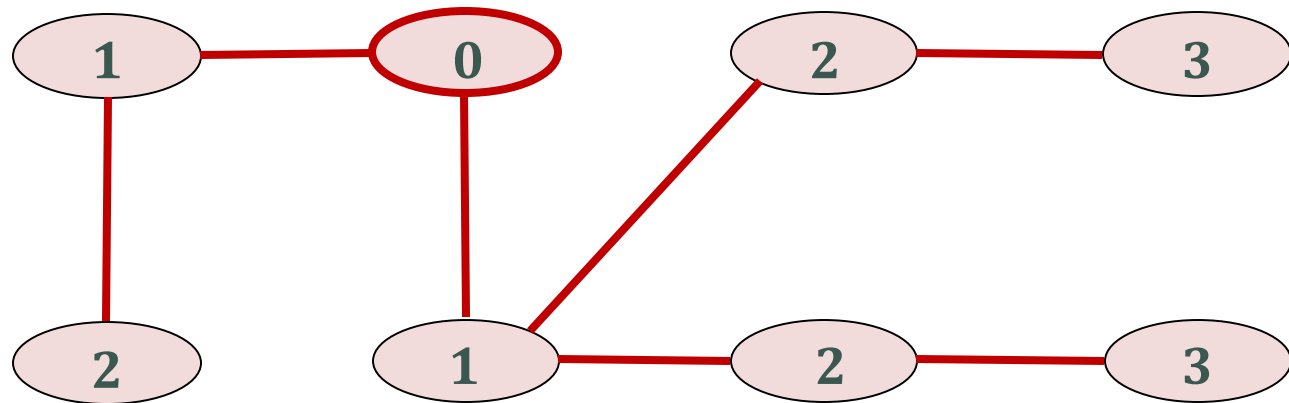
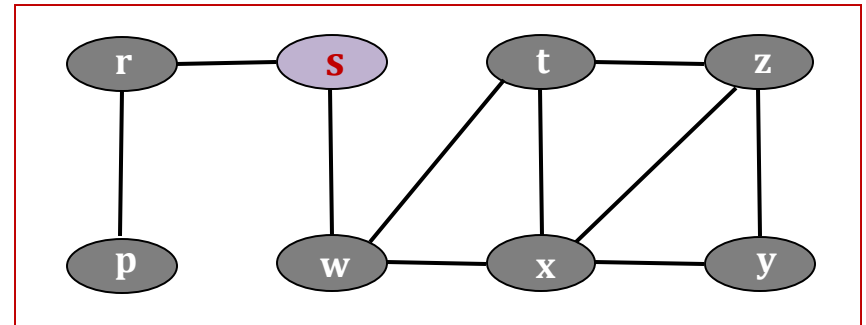
Εξερεύνηση κατά Πλάτος - BFS

BFS-δένδρο (δάσος)



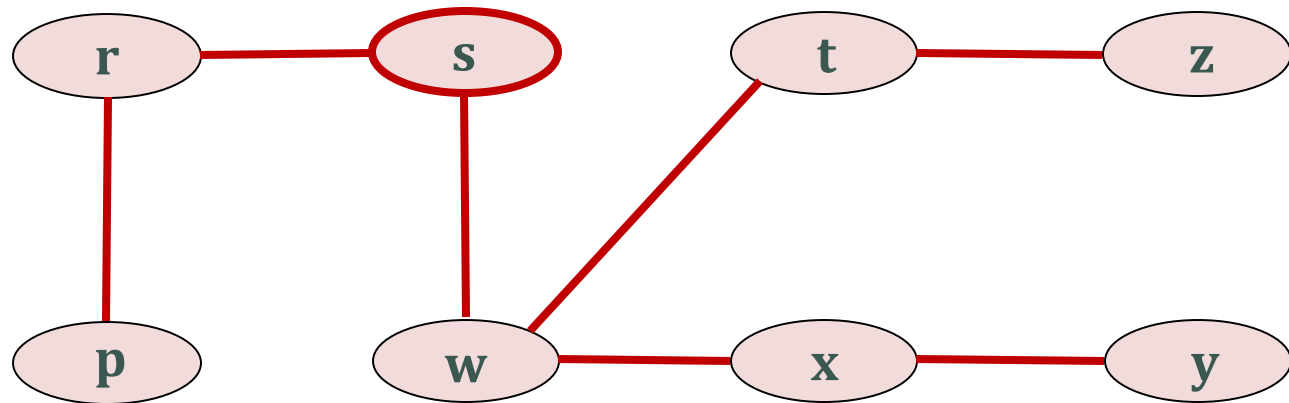
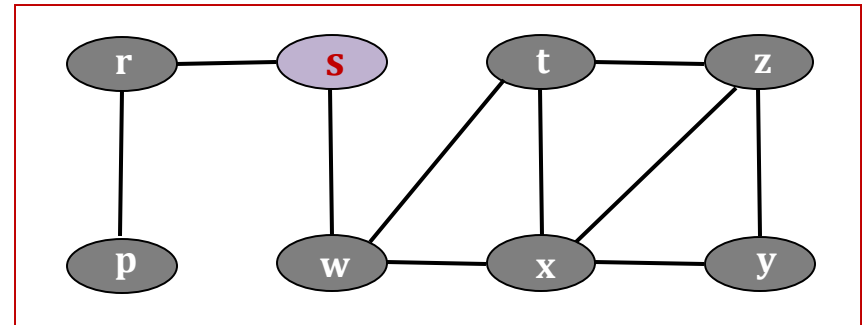
Εξερεύνηση κατά Πλάτος - BFS

BFS-δένδρο (δάσος)



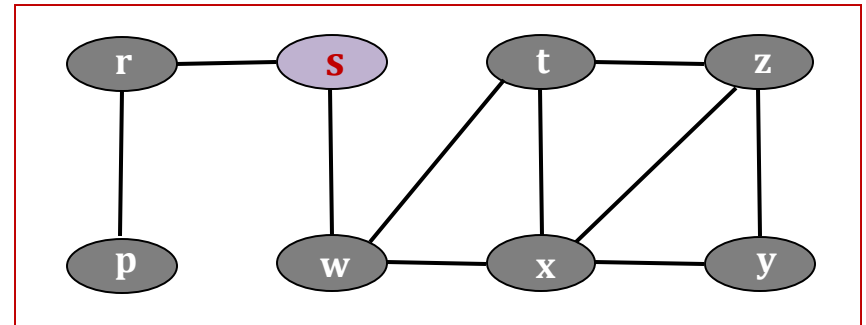
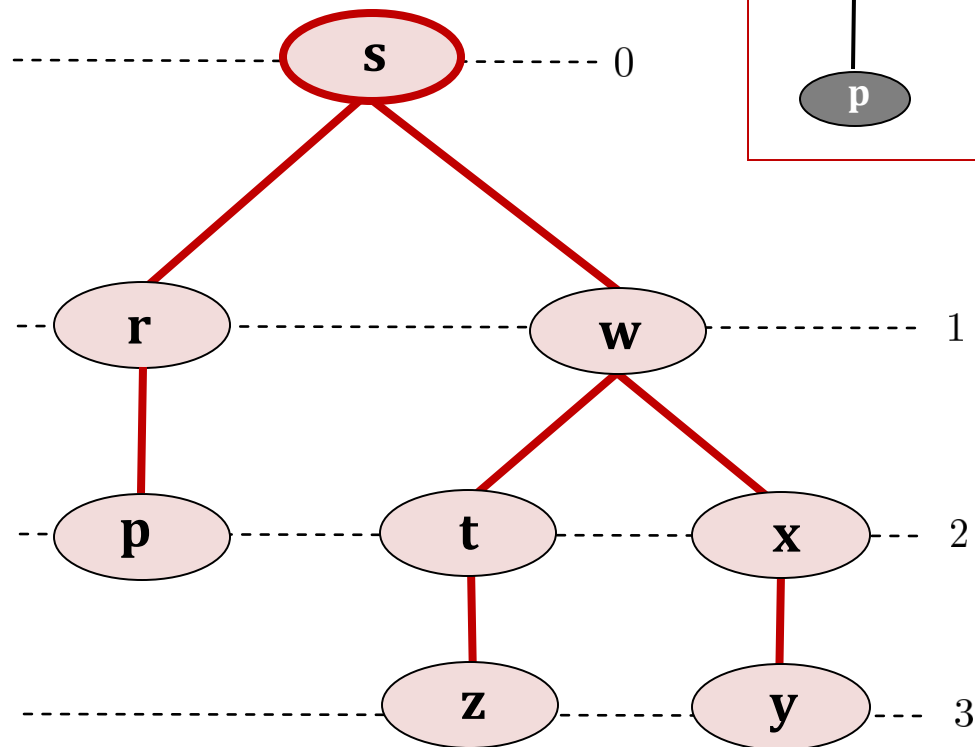
Εξερεύνηση κατά Πλάτος - BFS

BFS-δένδρο (δάσος)



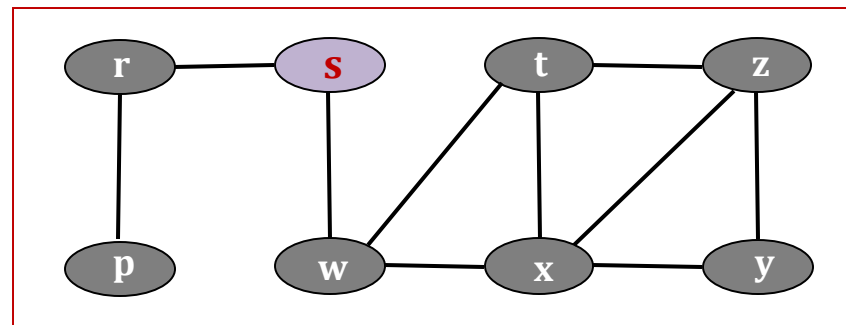
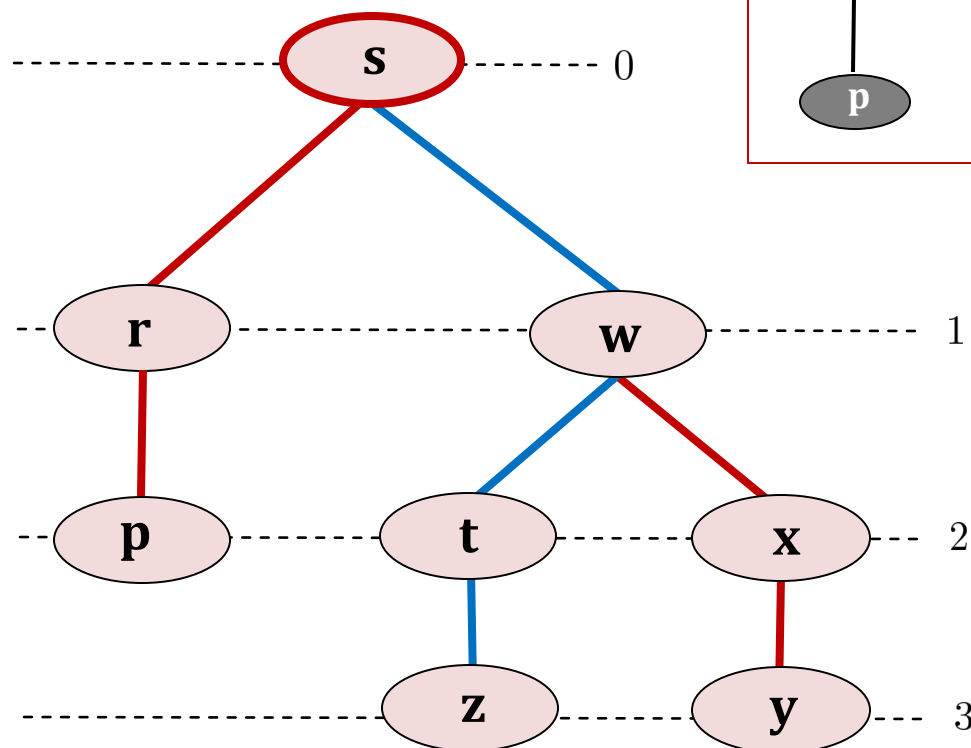
Εξερεύνηση κατά Πλάτος - BFS

BFS-δένδρο (δάσος)



Εφαρμογές BFS: συντομότερες διαδρομές

BFS-δένδρο (δάσος)

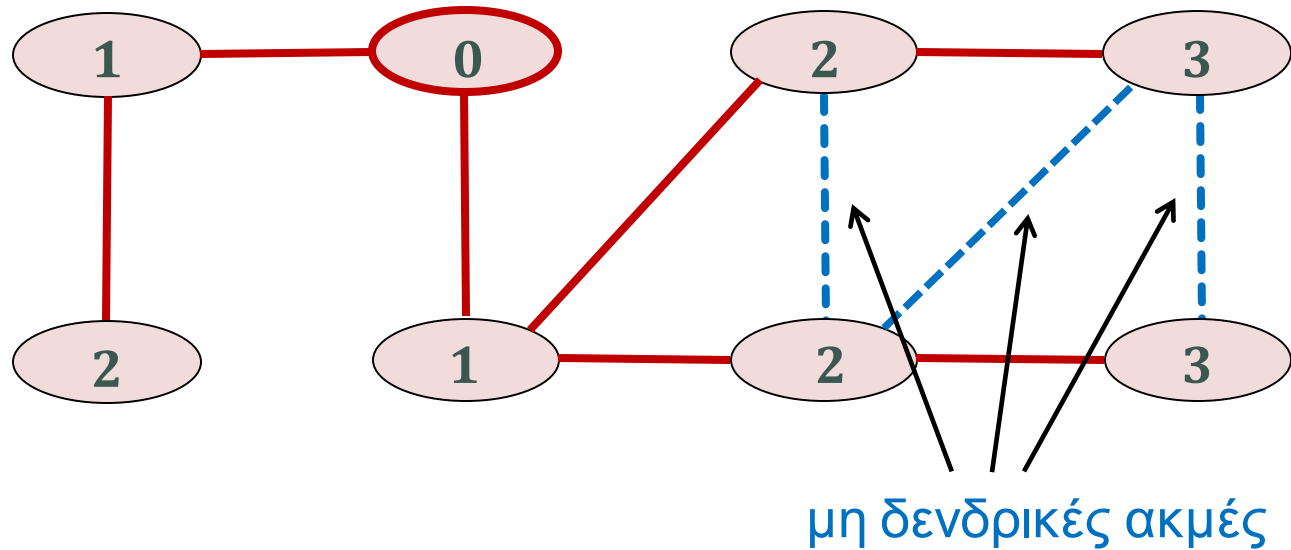
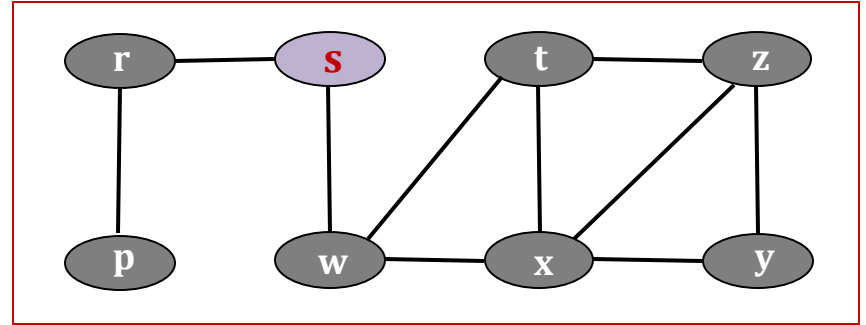


$$d(s, z) = 3$$

συντομότερη
διαδρομή $s \rightarrow z$:

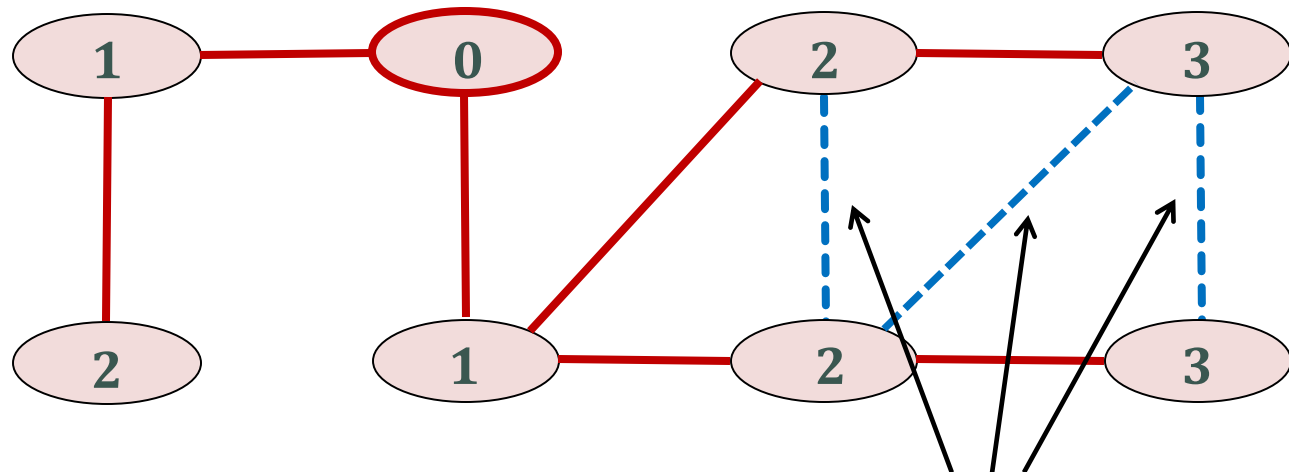
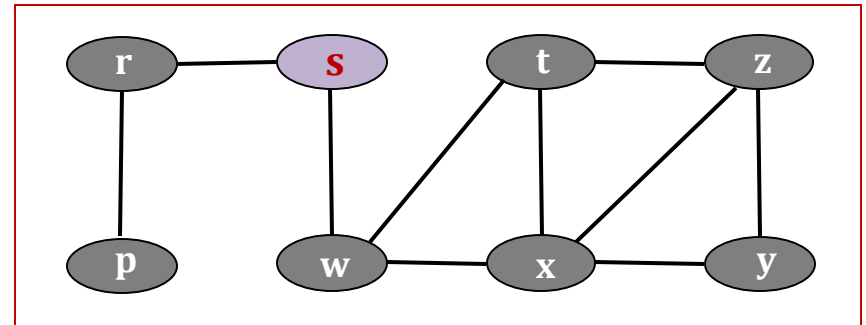
s w t z

Εφαρμογές BFS: εντοπισμός κύκλων

 BFS-δένδρο (δάσος)

Εφαρμογές BFS: εντοπισμός κύκλων

BFS-δένδρο (δάσος)

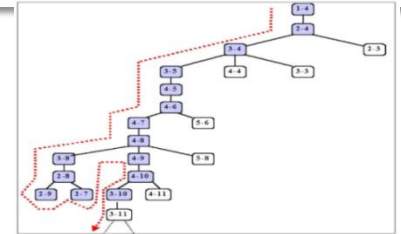


Αποστάσεις από κόμβο s

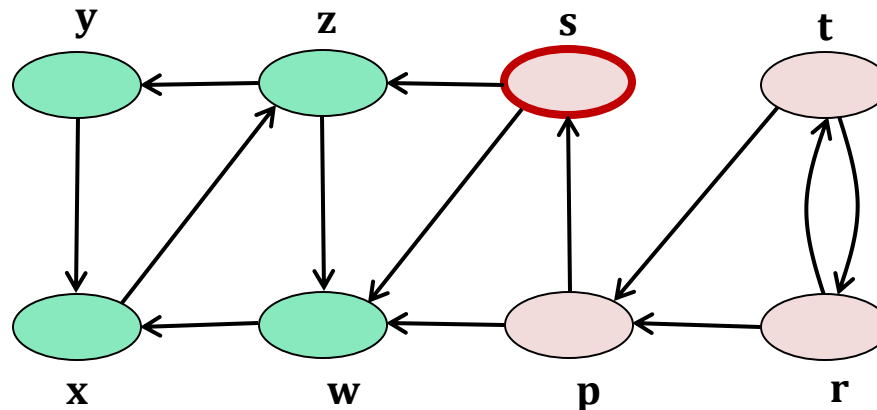
μη δενδρικές ακμές

Εξερεύνηση κατά Βάθος - DFS

■ Βασική Ιδέα DFS



- Η DFS προσπαθεί να εξερευνήσει πρώτα κόμβους σε όσο γίνεται **μεγαλύτερη απόσταση («βάθος»)** από έναν αρχικό κόμβο $s \in V$

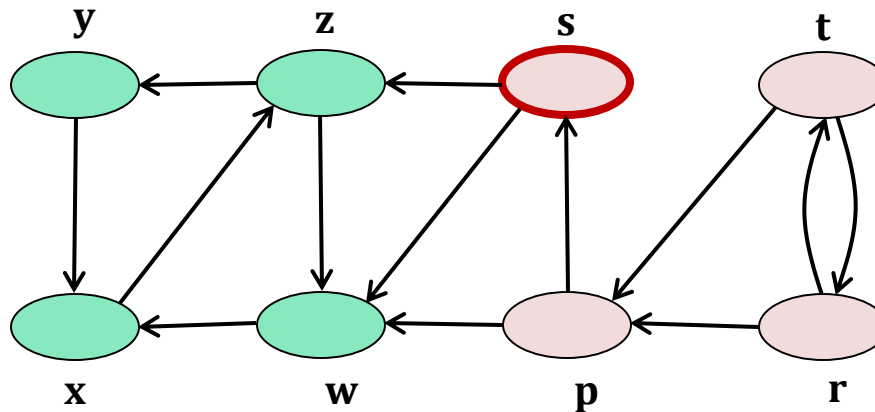


- Εάν $\exists$ **γείτονες** του s που **δεν έχουν εξερευνηθεί**, τότε επιλέγει έναν από αυτούς και συνεχίζει την εξερεύνηση από εκεί **αναδρομικά**

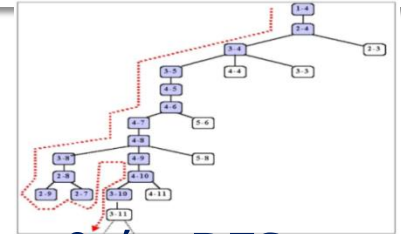
Εξερεύνηση κατά Βάθος - DFS

■ Βασική Ιδέα DFS

- Όταν **όλοι** οι γείτονες ενός κόμβου **v** έχουν εξερευνηθεί, η DFS «επιστρέφει» στον **γονέα u** του v, και συνεχίζει σε τυχόν ανεξερεύνητους γείτονες του **u**

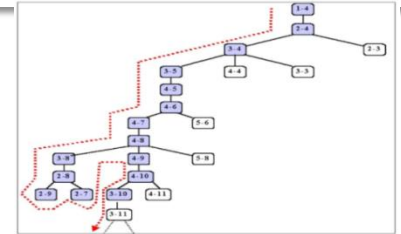


- Η αναδρομή της **DFS** **σταματά** όταν εξερευνηθούν **όλοι** οι προσβάσιμοι από τον **s** κόμβοι **x** του G : **s** → ... → **x**

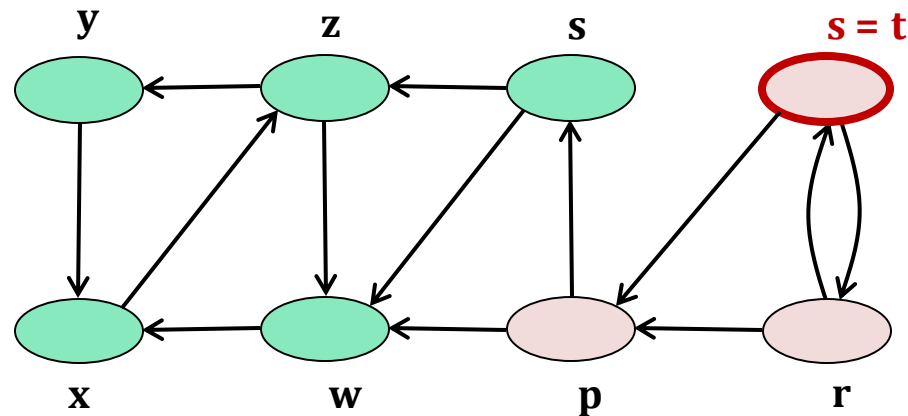


Εξερεύνηση κατά Βάθος - DFS

■ Βασική Ιδέα DFS



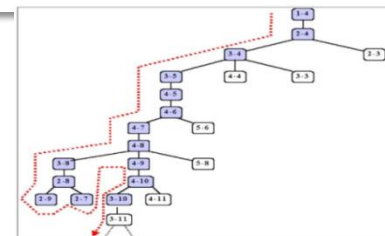
- Η αναδρομή της **DFS** **σταματά** όταν εξερευνηθούν **όλοι οι προσβάσιμοι από τον s κόμβοι x** του $G : s \rightarrow \dots \rightarrow x$



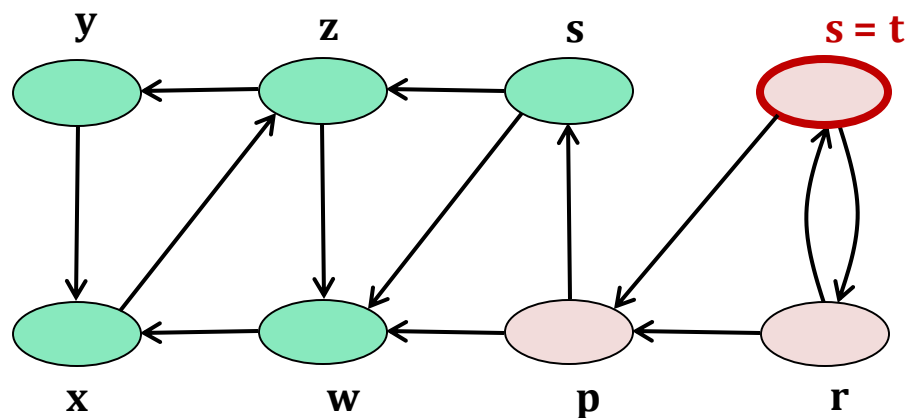
- Εάν $\exists$ κόμβοι που **δεν έχουν εξερευνηθεί**, τότε επιλέγεται νέος μη-εξερευνημένος $s' \in V$ και τίθεται ως νέος **κόμβος εκκίνησης**: $s \leftarrow s'$

Εξερεύνηση κατά Βάθος - DFS

Βασική Ιδέα DFS



- Η αναδρομή της **DFS** **σταματά** όταν εξερευνηθούν **όλοι οι προσβάσιμοι από τον s κόμβοι x** του G : $s \rightarrow \dots \rightarrow x$


$$a(x)/t(x)$$

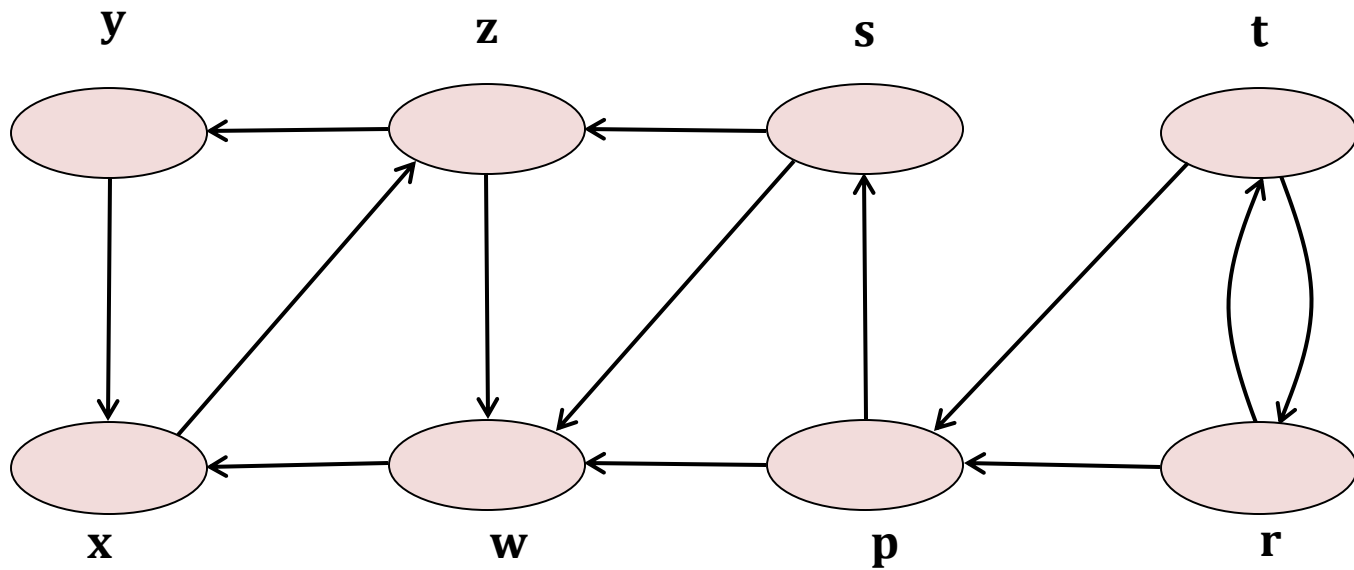
$a(x)$ χρόνος **πρώτης**
 επίσκεψης, και
 $t(x)$ χρόνος **τελευταίας**
 επίσκεψης του x

- Εάν $\exists$ κόμβοι που **δεν έχουν εξερευνηθεί**, τότε επιλέγεται νέος μη-εξερευνημένος $s' \in V$ και τίθεται ως νέος **κόμβος εκκίνησης**:
 $s \leftarrow s'$

Εξερεύνηση κατά Βάθος - DFS



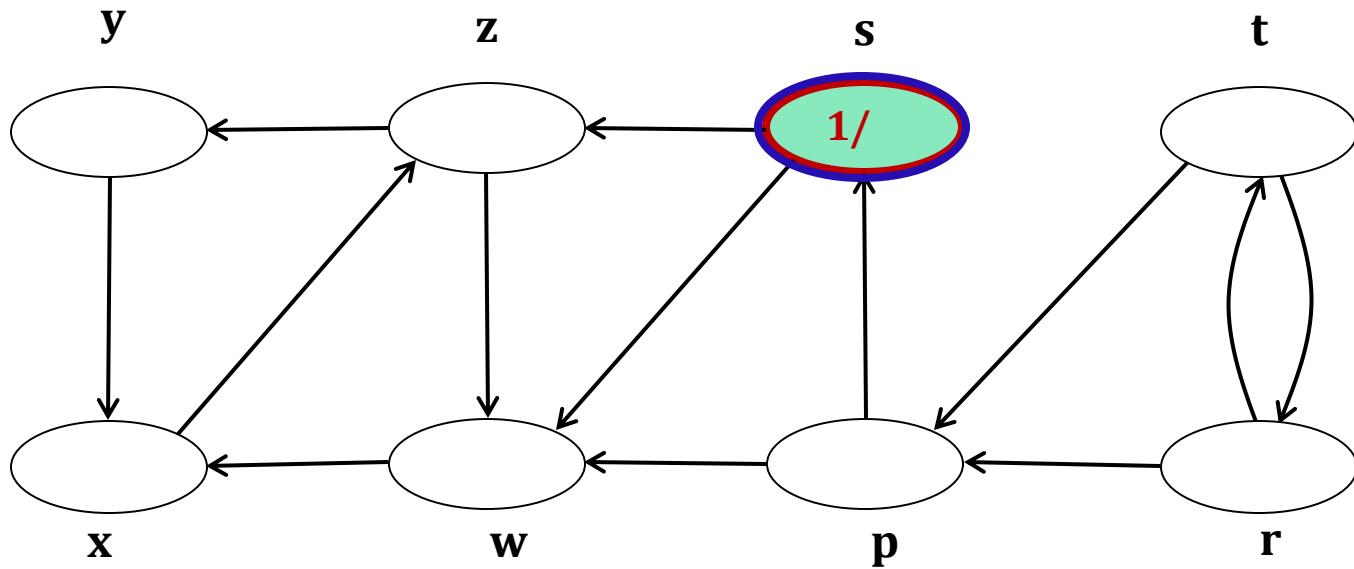
Παράδειγμα



Εξερεύνηση κατά Βάθος - DFS



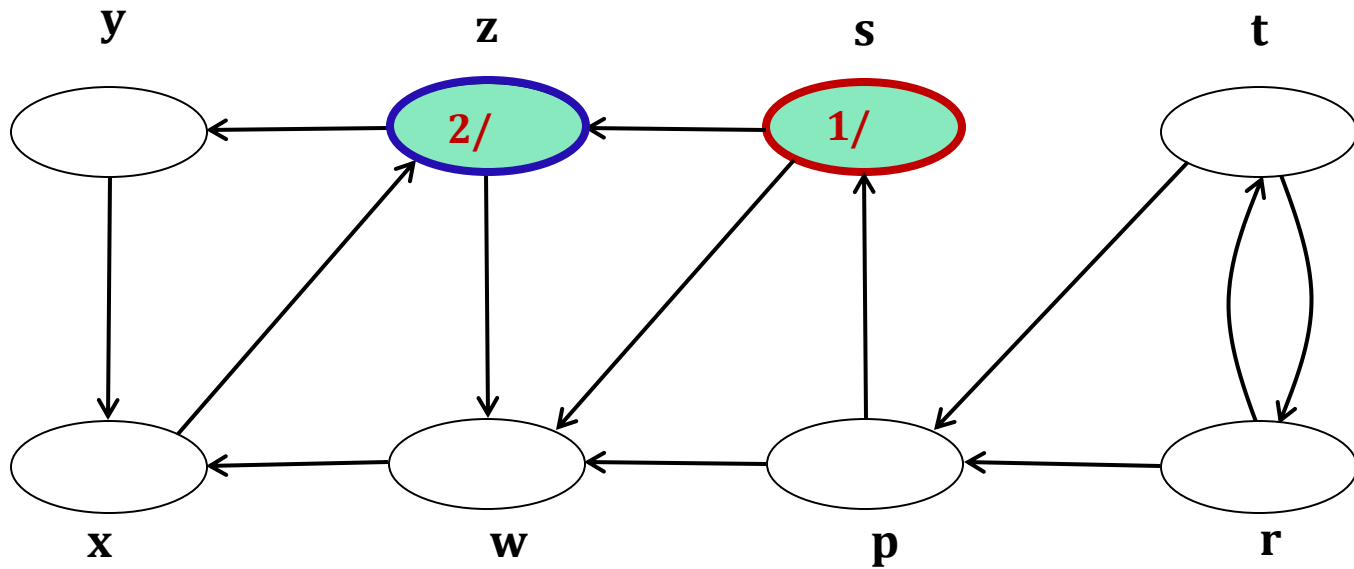
Παράδειγμα



Εξερεύνηση κατά Βάθος - DFS

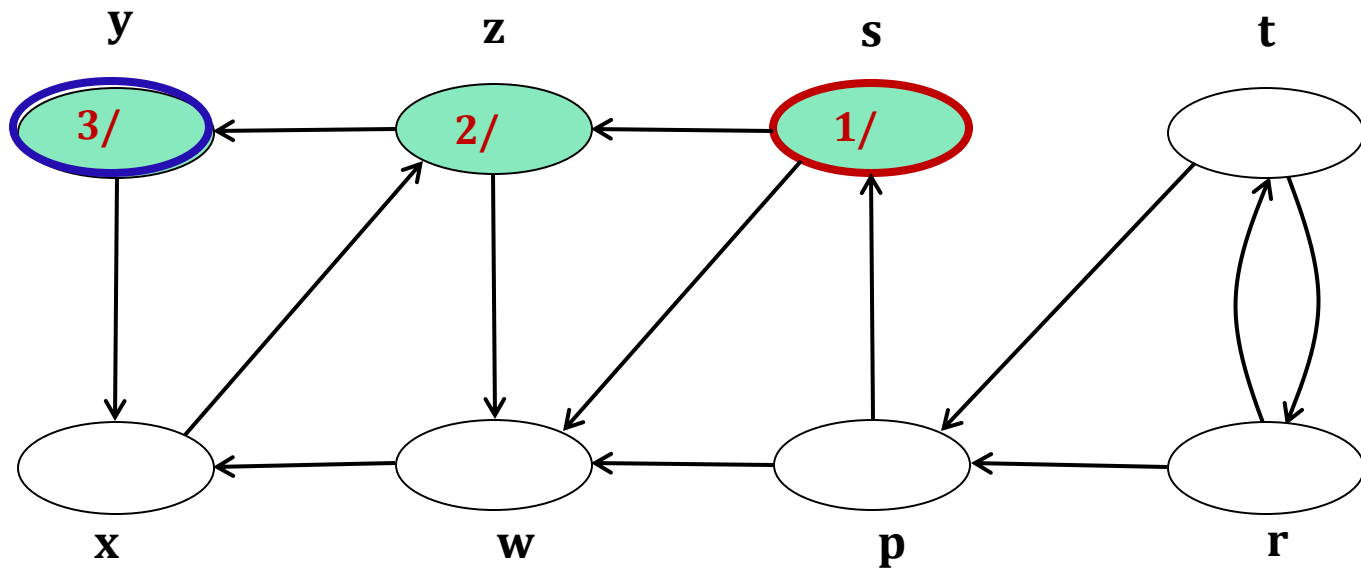


Παράδειγμα



Εξερεύνηση κατά Βάθος - DFS

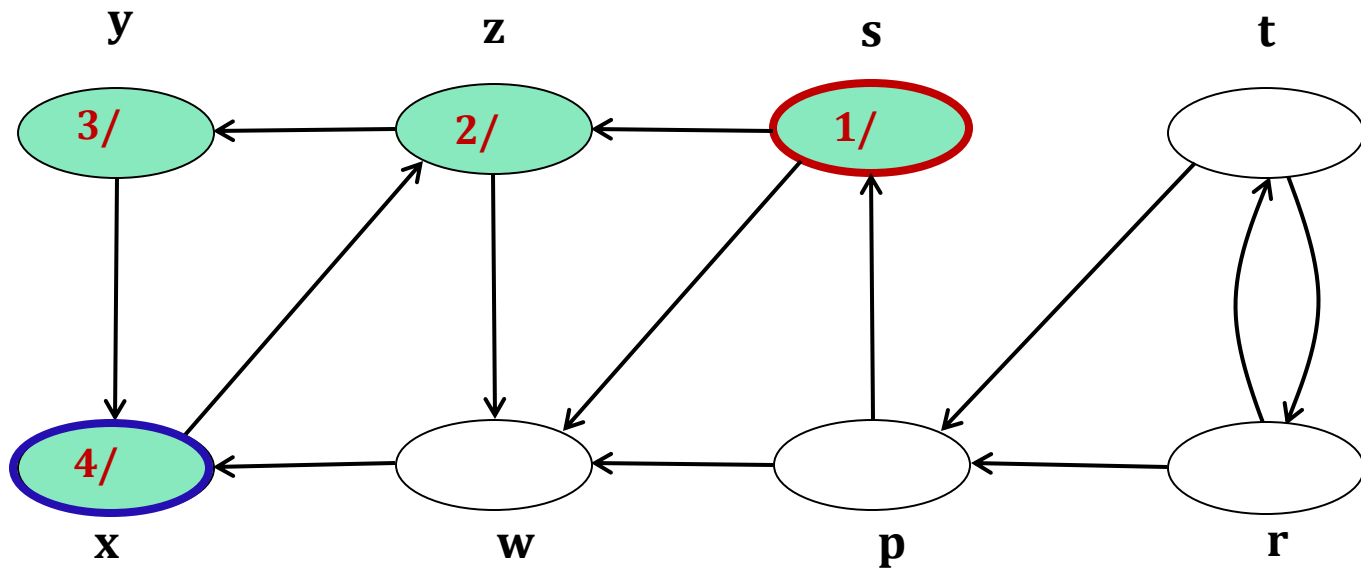
Παράδειγμα



Εξερεύνηση κατά Βάθος - DFS



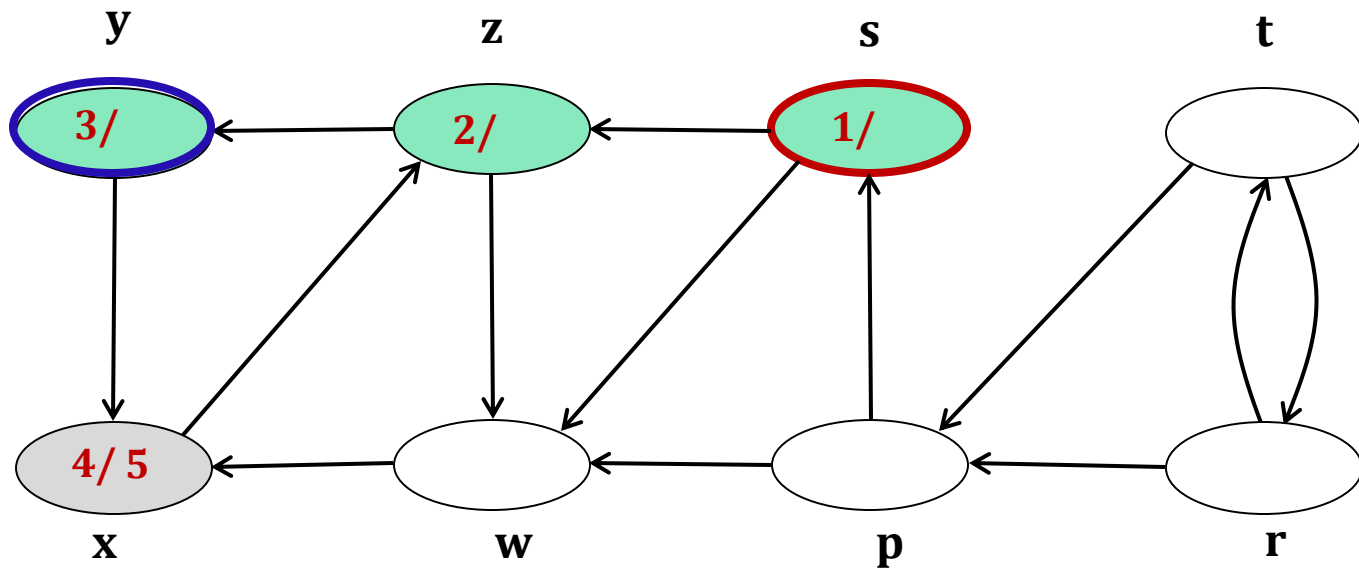
Παράδειγμα



Εξερεύνηση κατά Βάθος - DFS



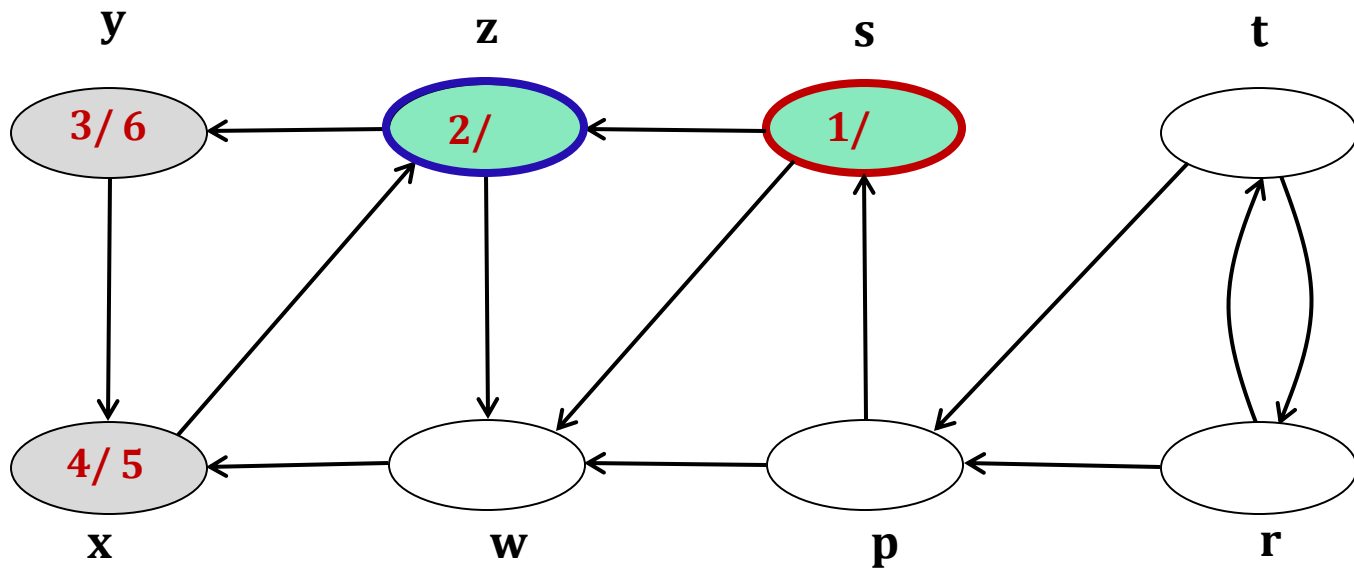
Παράδειγμα



Εξερεύνηση κατά Βάθος - DFS

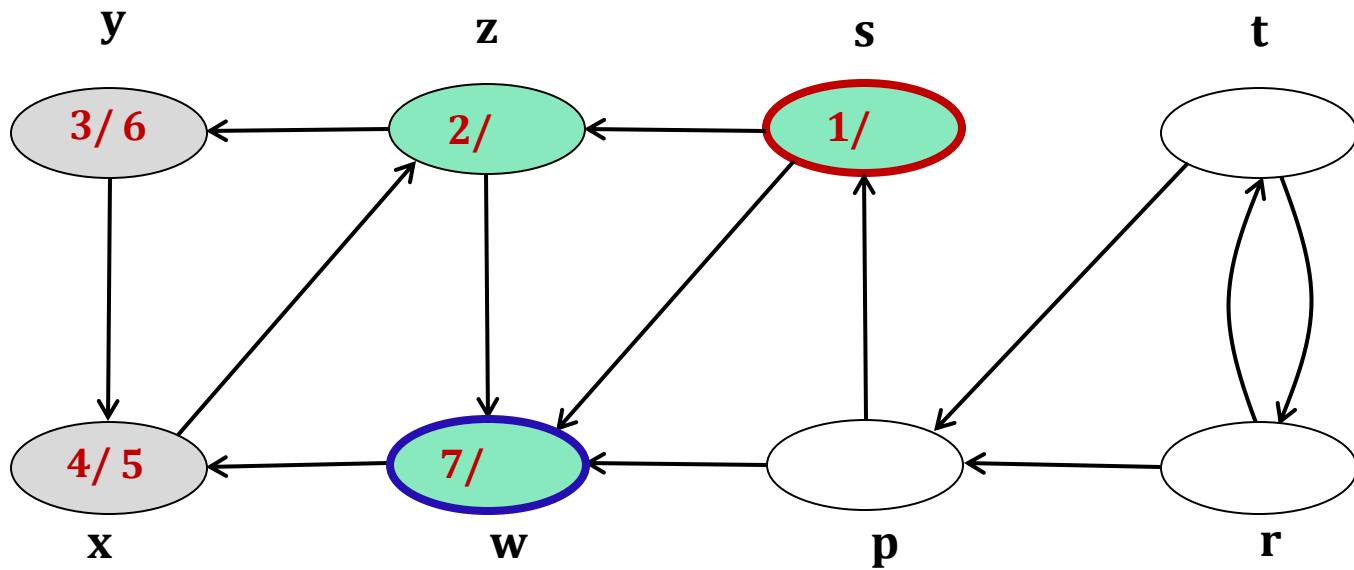


Παράδειγμα



Εξερεύνηση κατά Βάθος - DFS

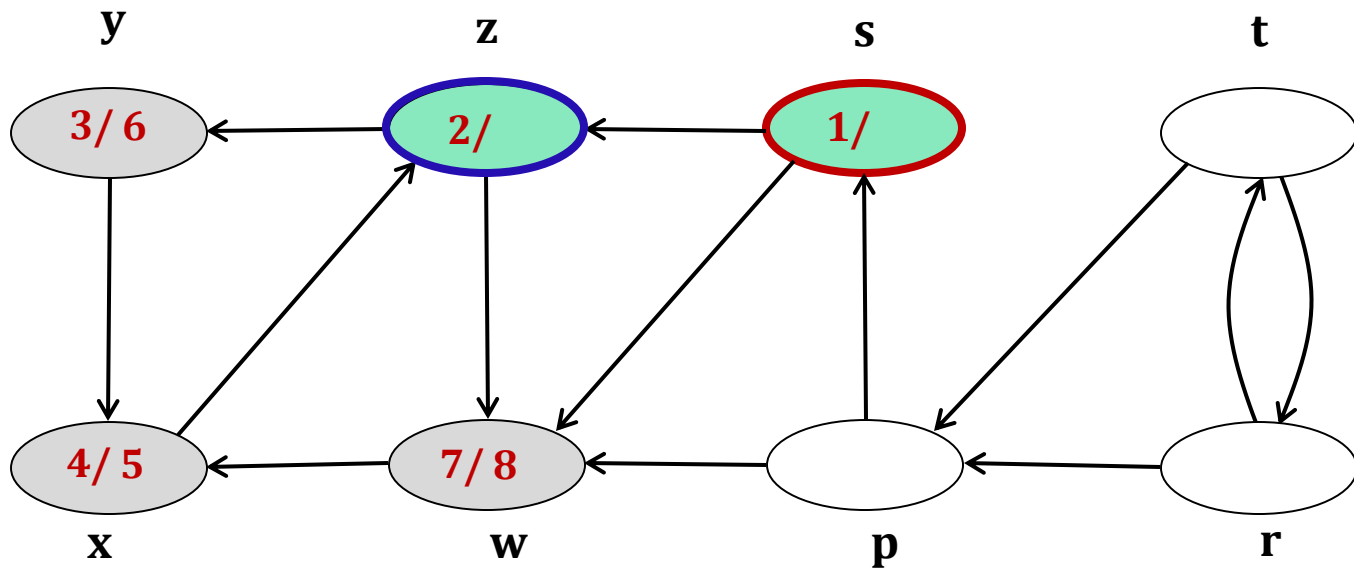
Παράδειγμα



Εξερεύνηση κατά Βάθος - DFS

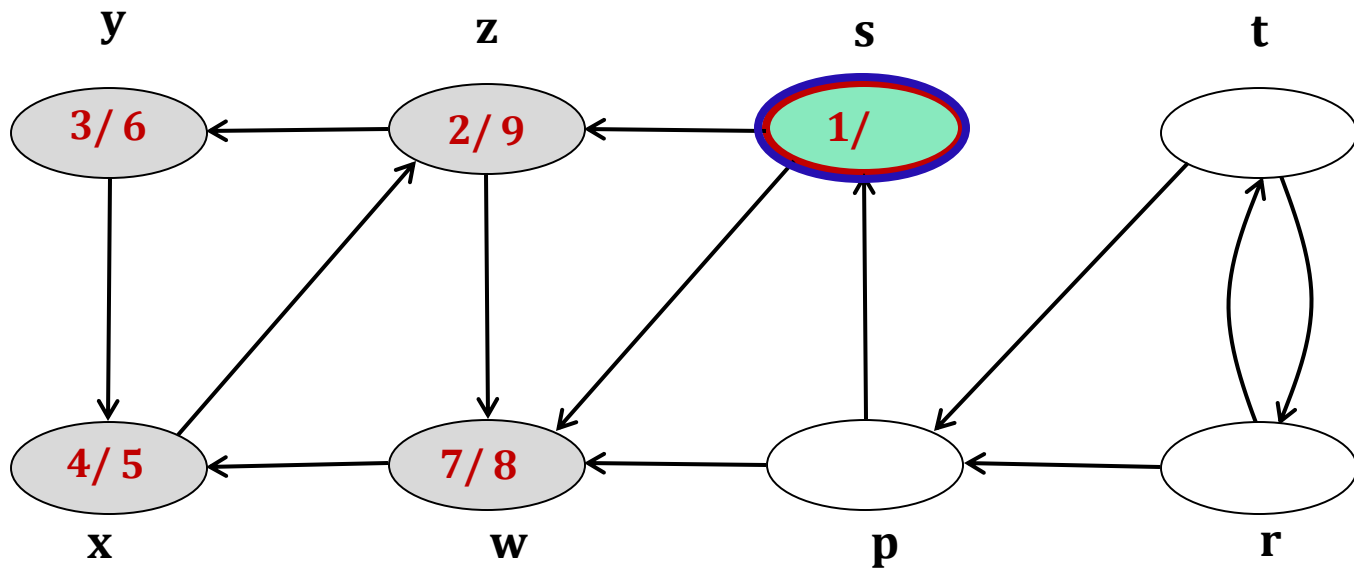


Παράδειγμα



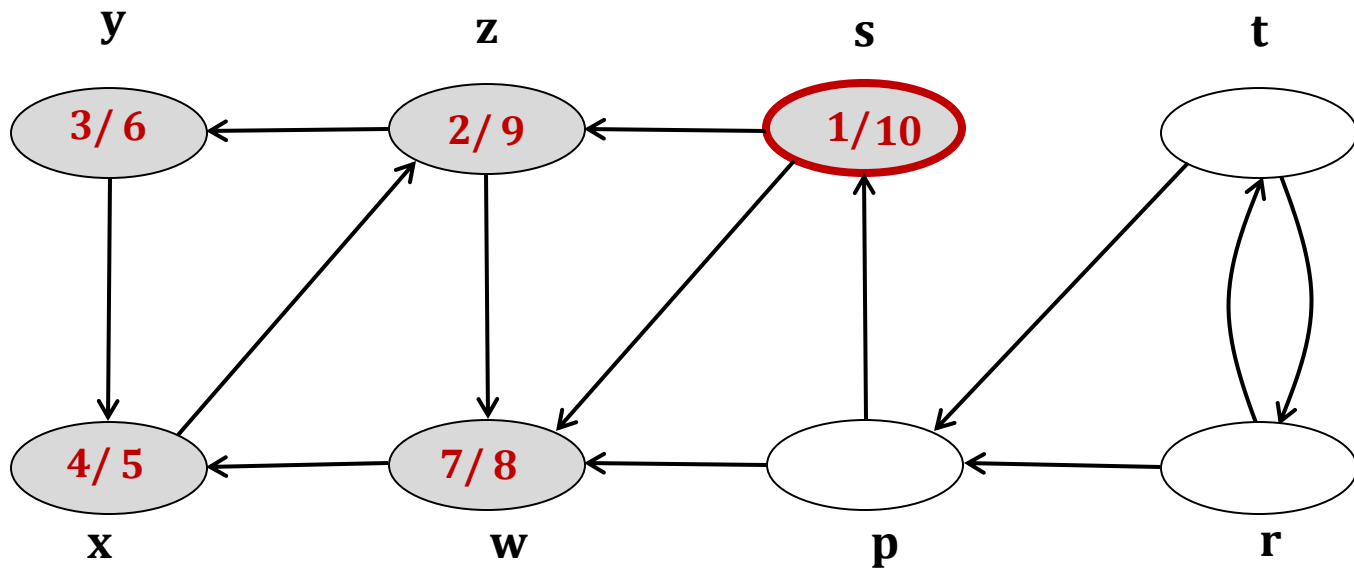
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



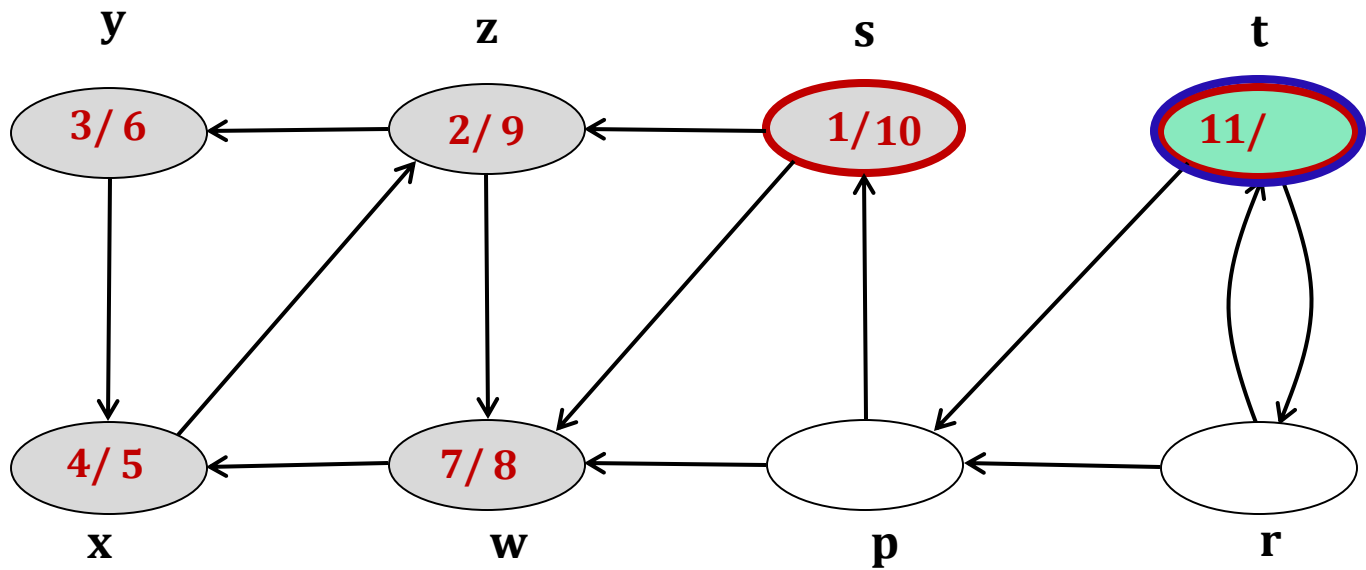
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



Εξερεύνηση κατά Βάθος - DFS

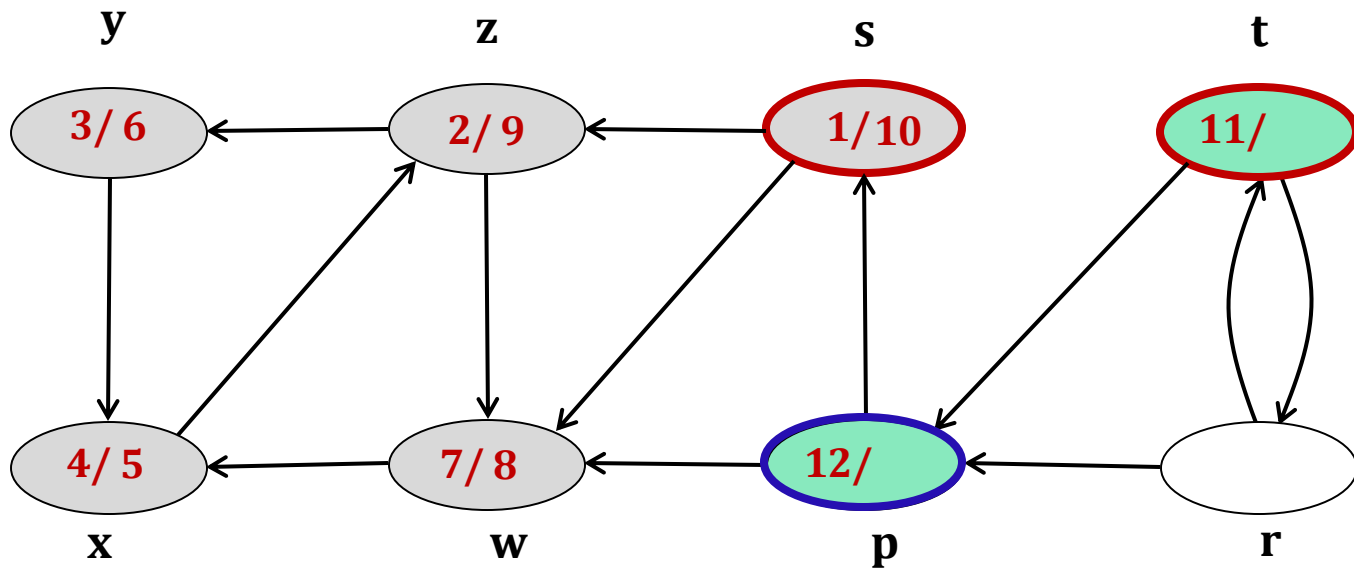
Παράδειγμα



Εξερεύνηση κατά Βάθος - DFS



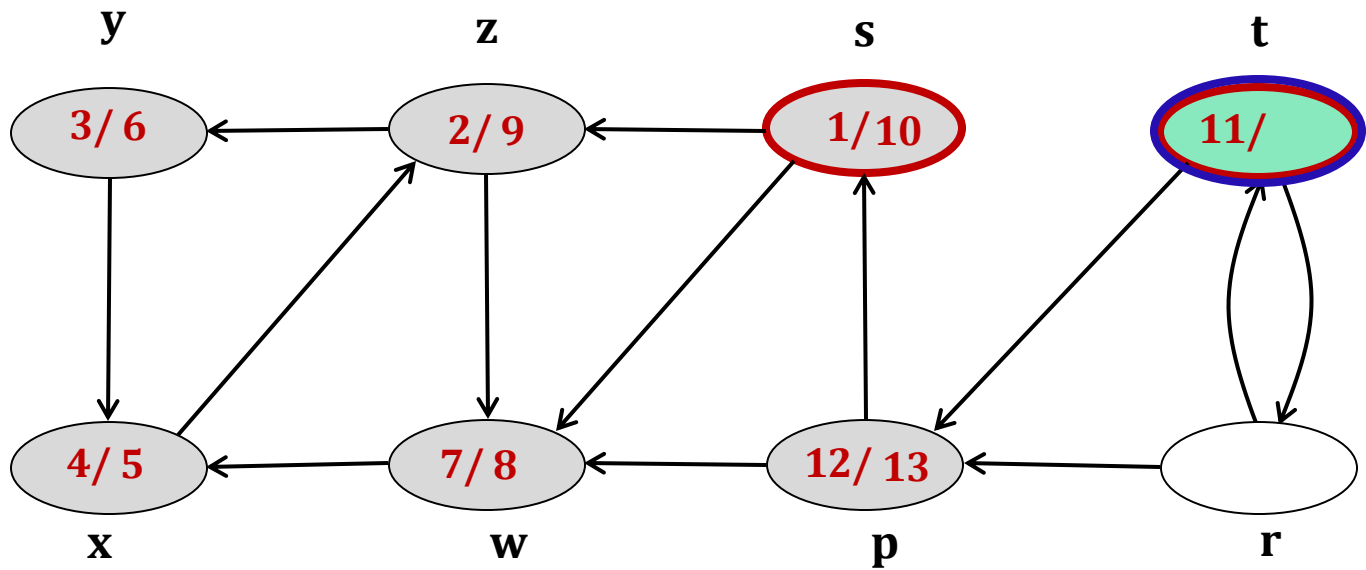
Παράδειγμα



Εξερεύνηση κατά Βάθος - DFS

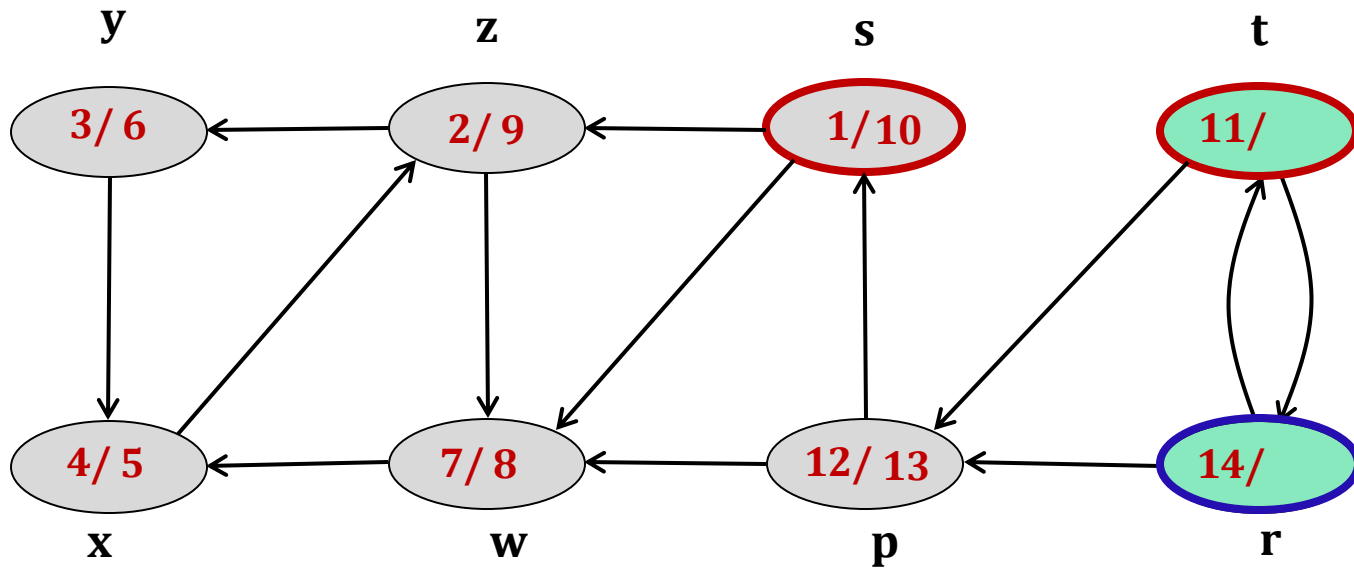


Παράδειγμα



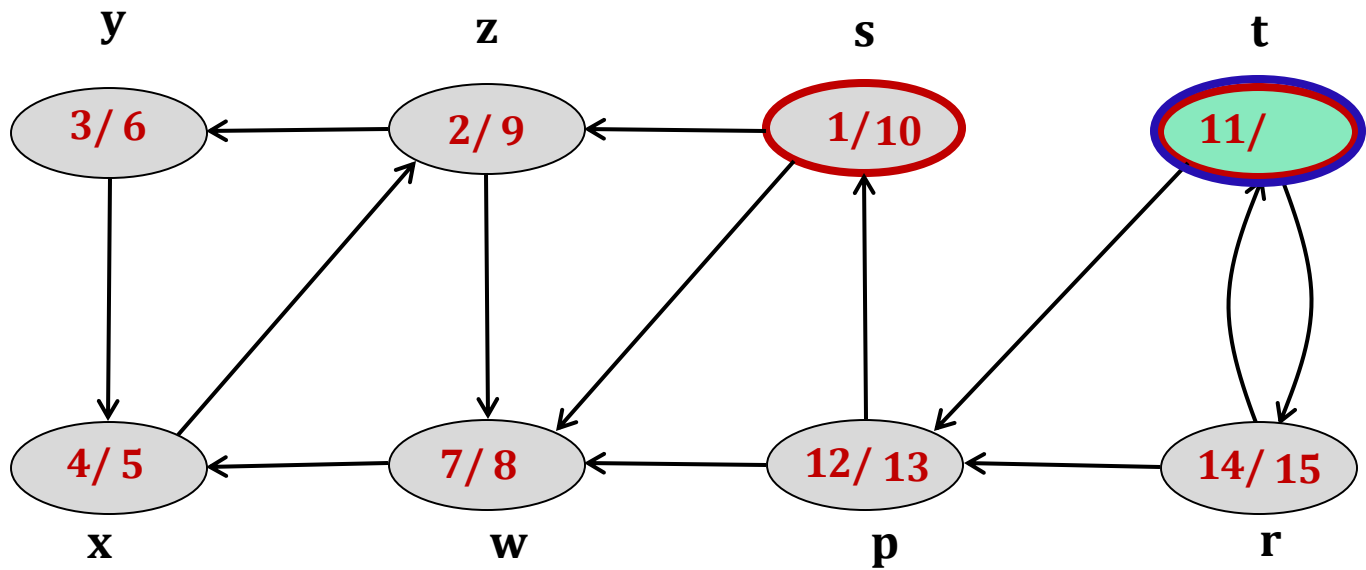
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



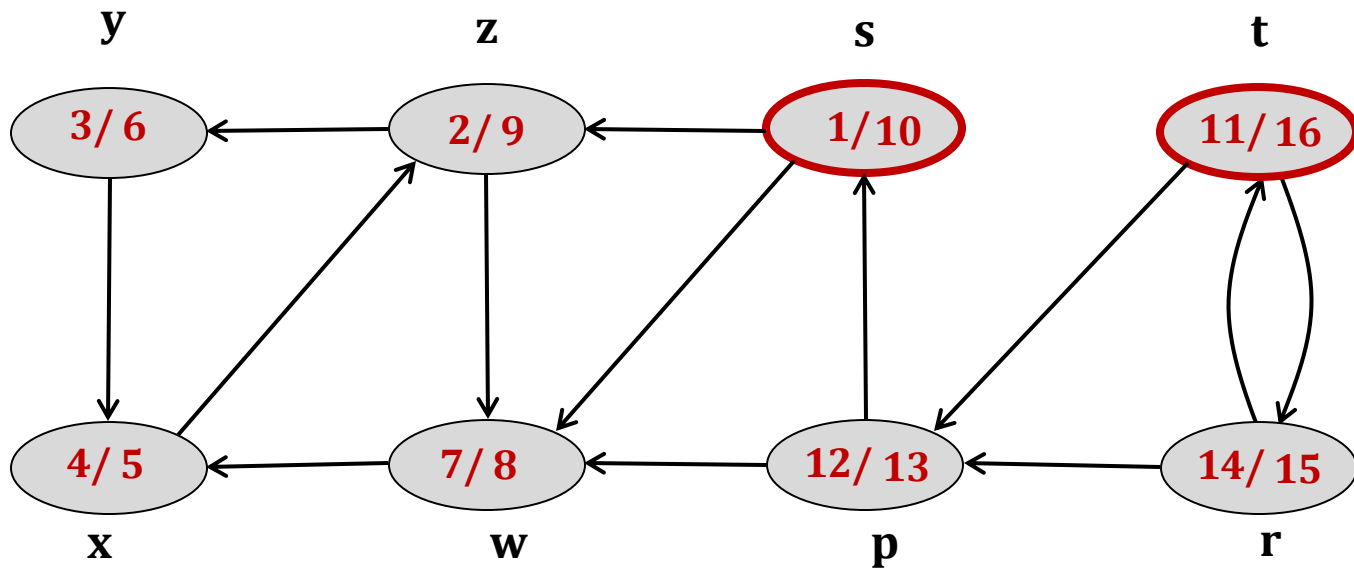
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



έξοδος DFS = $\{(s, z, y, x, w), (t, p, r)\}$

εμφάνιση κατά σειρά πρώτης επίσκεψης

Εξερεύνηση κατά Βάθος – DFS: χωρίς χρόνους επίσκεψης



DFS($G=(V,E)$)

- 1. for each** vertex $v \in V$ **do**
2. status[v] $\leftarrow$ UNVISITED ; $\pi(v) \leftarrow \text{nil}$
3. time $\leftarrow 0$
- 4. while** $\exists s$ with status[s] = UNVISITED **do**
5. DFS-Help(s)

DFS-Help(u)

status[u] $\leftarrow$ VISITED

A[u] $\leftarrow$ time $\leftarrow$ time+1

for each vertex $v \in \text{adj}(u)$ **do**

if status[v] = UNVISITED

$\pi(v) \leftarrow u$; DFS-Help(v)

status[u] $\leftarrow$ EXPLORED

T[u] $\leftarrow$ time $\leftarrow$ time+1

VISITED: η
εξερεύνηση
του κόμβου
ξεκίνησε

Η μορφή αυτή
αρκεί για
εύρεση όλων
προσβάσιμων
κόμβων

Εξερεύνηση κατά Βάθος – DFS: με χρόνους επίσκεψης



DFS($G=(V,E)$)

- 1. for each** vertex $v \in V$ **do**
2. $\text{status}[v] \leftarrow \text{UNVISITED}$; $\pi(v) \leftarrow \text{nil}$
3. $\text{time} \leftarrow 0$
- 4. while** $\exists s$ with $\text{status}[s] = \text{UNVISITED}$ **do**
5. $\text{DFS-Help}(s)$

DFS-Help(u)

$\text{status}[u] \leftarrow \text{VISITED}$
 $A[u] \leftarrow \text{time} \leftarrow \text{time}+1$
for each vertex $v \in \text{adj}(u)$ **do**
 if $\text{status}[v] = \text{UNVISITED}$
 $\pi(v) \leftarrow u$; $\text{DFS-Help}(v)$
 $\text{status}[u] \leftarrow \text{EXPLORED}$
 $T[u] \leftarrow \text{time} \leftarrow \text{time}+1$

VISITED: η
εξερεύνηση
του κόμβου
ξεκίνησε

EXPLORED: η
εξερεύνηση
του κόμβου
έχει
ολοκληρωθεί
(συχνά δεν
χρειάζεται και
παραλείπεται)

Εξερεύνηση κατά Βάθος - DFS

Πολυπλοκότητα DFS

✓ Τα βήματα 1 και 4 εκτελούνται σε χρόνο $O(|V|)$

✓ Το βήμα 3 σε $O(1)$

✓ Η διαδικασία DFS-Help() καλείται 1 φορά για $\forall v \in V$

✓ Η DFS-Help(u) εκτελεί $\deg(u)=|\text{adj}(u)|$ ελέγχους.

$\sum_{u \in V} \deg(u) = 2|E| = O(|E|)$ (μη κατ.)

$\sum_{u \in V} \deg(u) = |E| = O(|E|)$ (κατευθ.)

✓ Άρα, πολυπλοκότητα DFS:

$$O(|V|+|E|)=O(n+m)$$

DFS($G=(V,E)$)

1. **for each** vertex $v \in V$ **do**
2. $\text{status}[v] \leftarrow \text{UNVISITED}$; $\pi(v) \leftarrow \text{nil}$
3. $\text{time} \leftarrow 0$
4. **while** $\exists s$ with $\text{status}[s] = \text{UNVISITED}$ **do**
5. DFS-Help(s)

DFS-Help(u)

$\text{status}[u] \leftarrow \text{VISITED}$

$A[u] \leftarrow \text{time} \leftarrow \text{time}+1$

for each vertex $v \in \text{adj}(u)$ **do**

if $\text{status}[v] = \text{UNVISITED}$

$\pi(v) \leftarrow u$; DFS-Help(v)

$\text{status}[u] \leftarrow \text{EXPLORED}$ // συχνά παραλείπεται

$T[u] \leftarrow \text{time} \leftarrow \text{time}+1$

Εξερεύνηση κατά Βάθος - DFS

Κατηγορίες ακμών κατά την DFS

T : Δενδρικές ακμές (**tree-edges**)

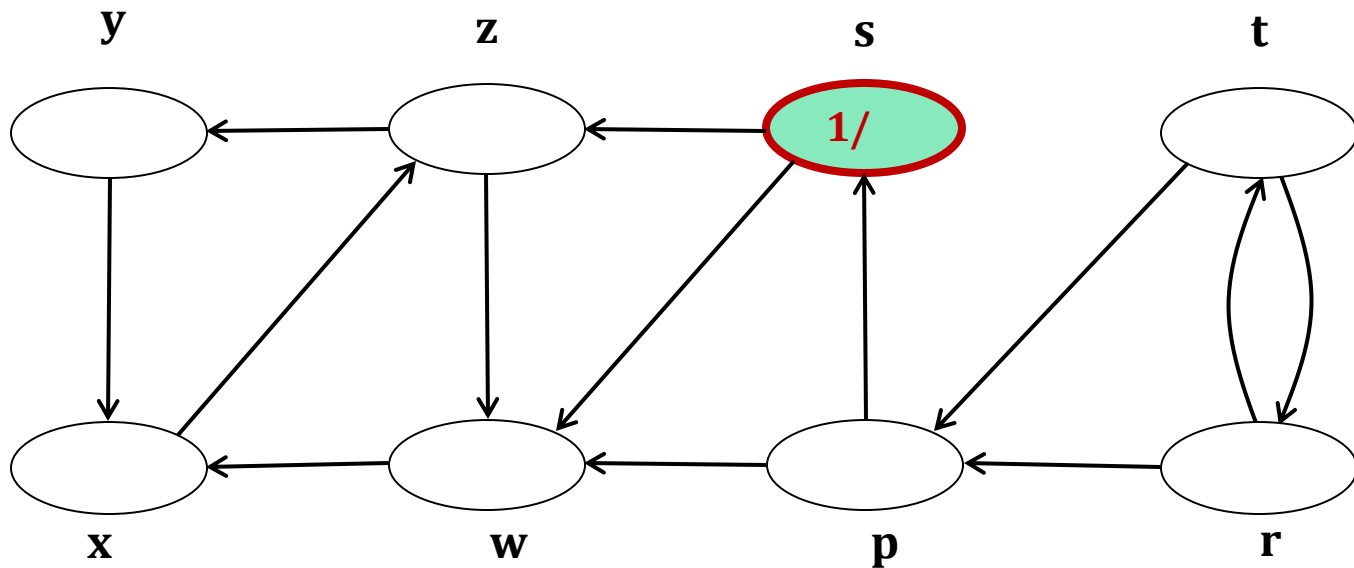
B : Οπισθο-ακμές ή ανιούσες (**back-edges**)

F : Εμπρόσθιες ακμές ή κατιούσες (**forward-edges**)

C : Διασχίζουσες ή εγκάρσιες ακμές (**cross-edges**)

Εξερεύνηση κατά Βάθος - DFS

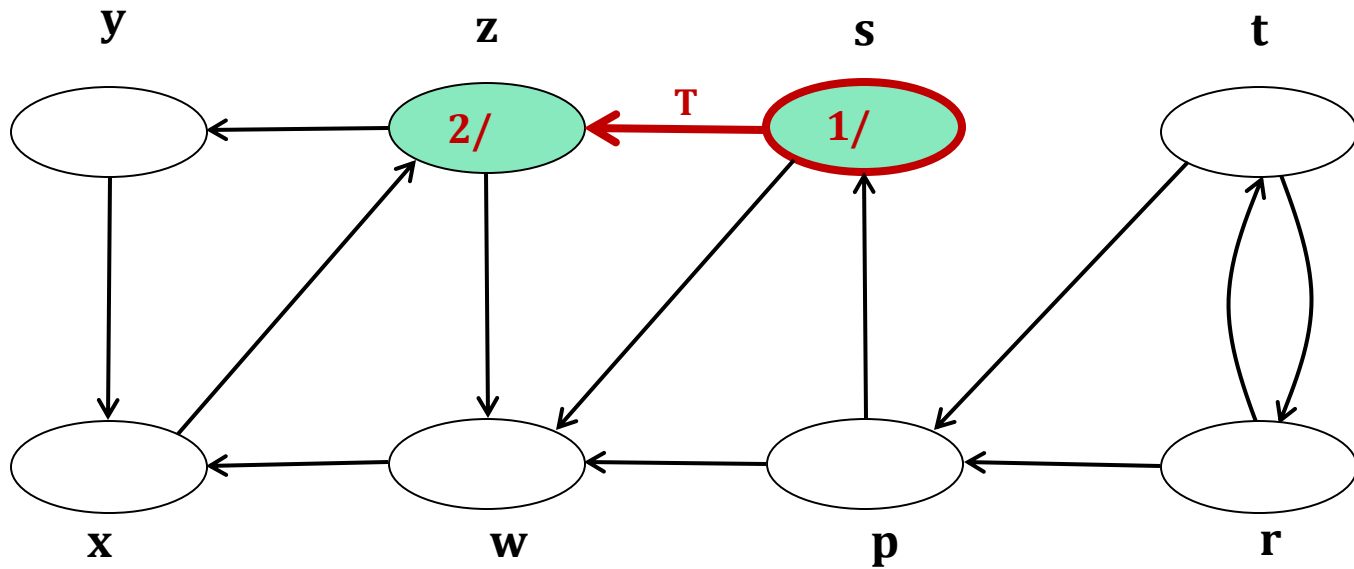
Παράδειγμα



Εξερεύνηση κατά Βάθος - DFS

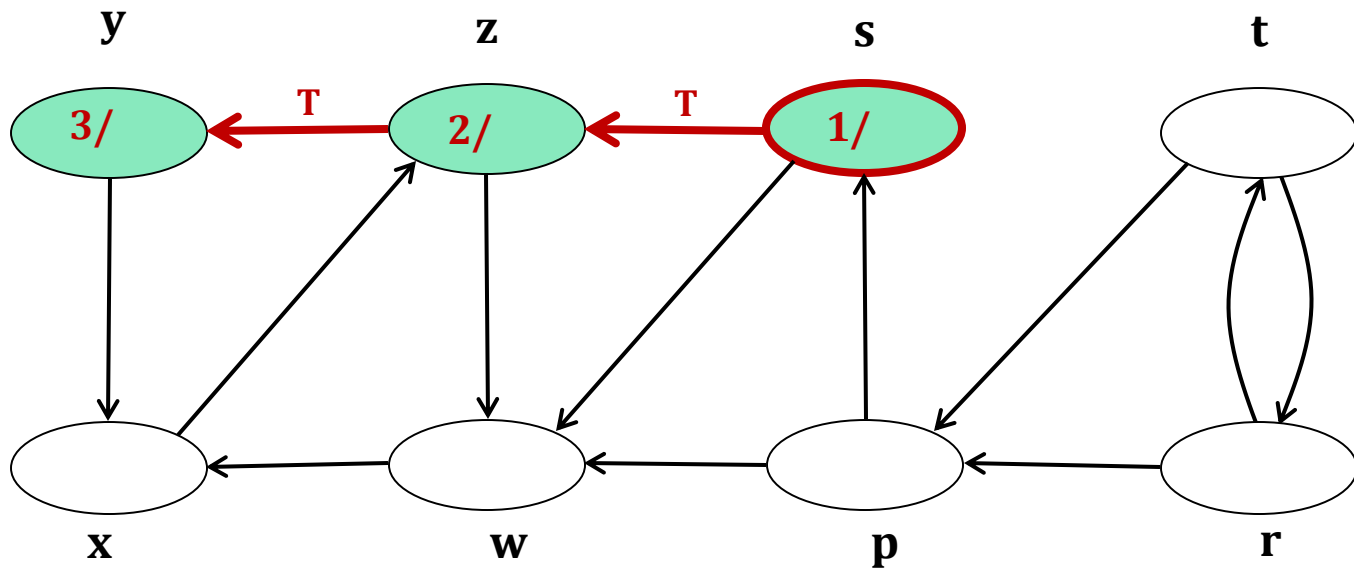


Παράδειγμα



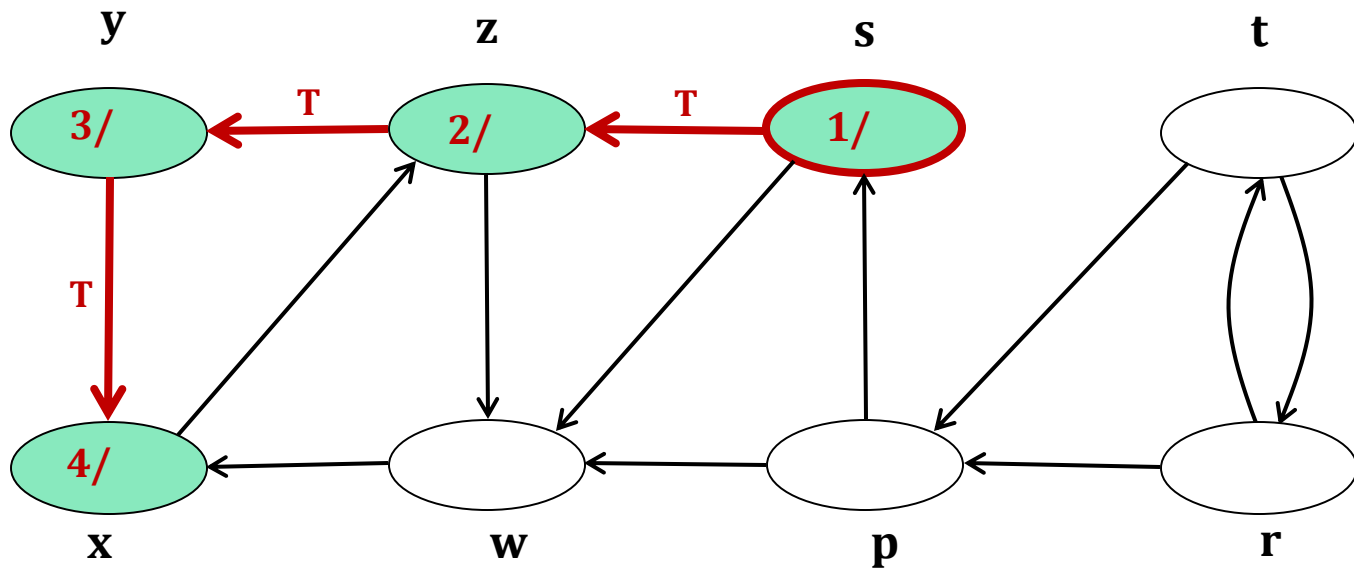
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



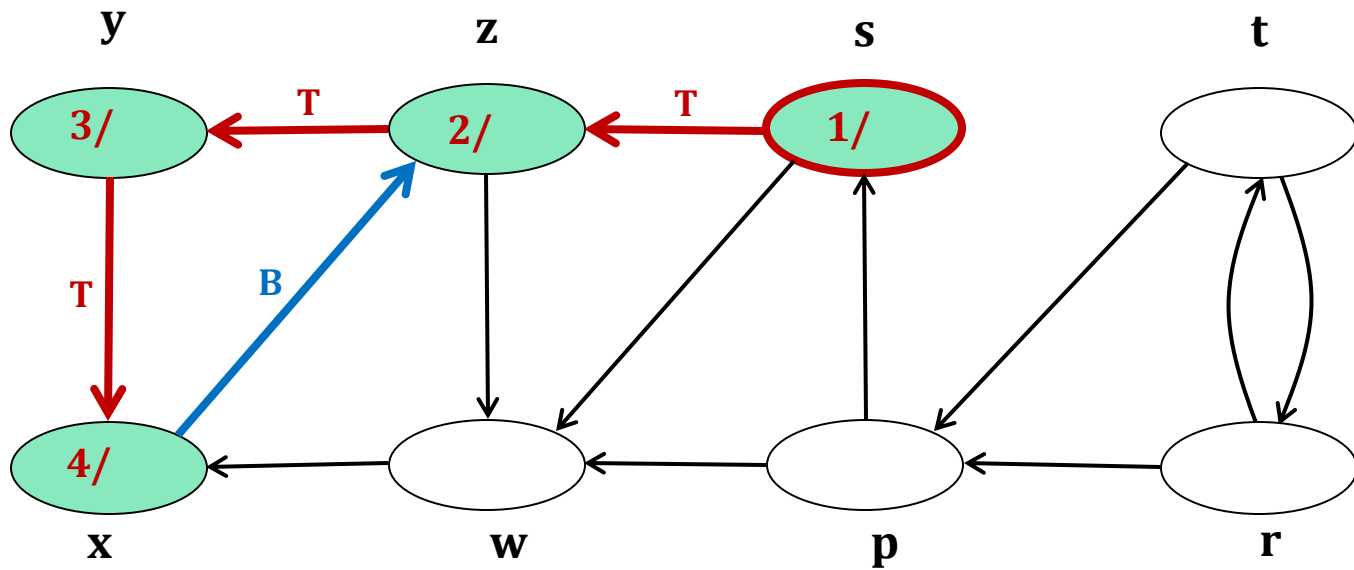
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



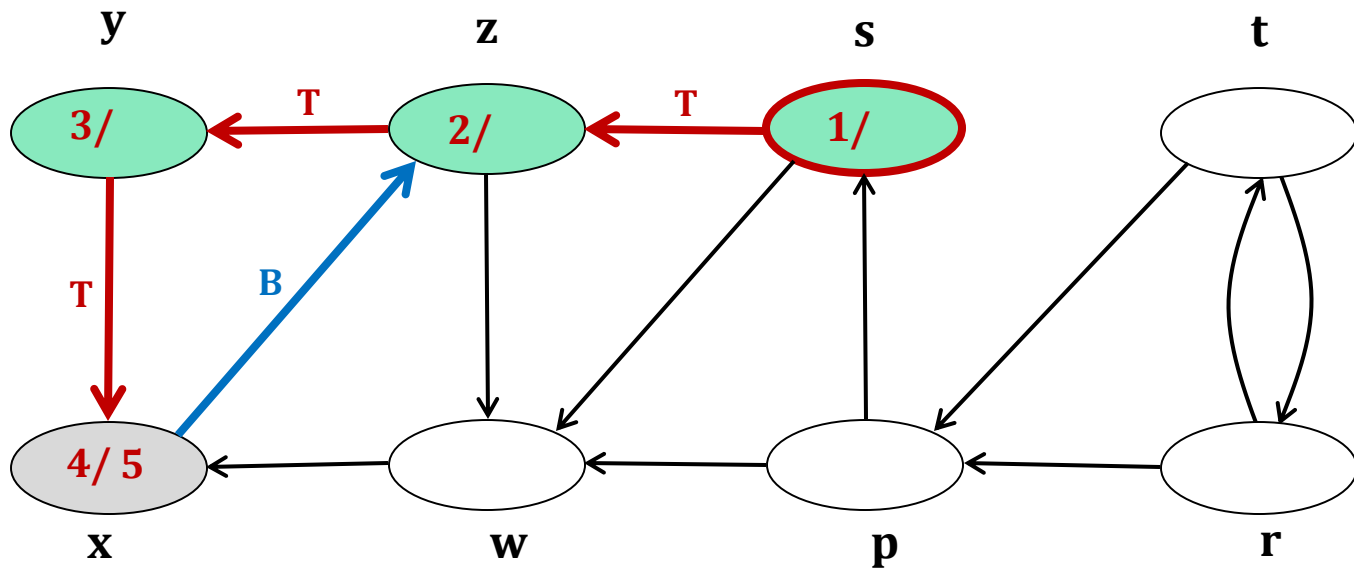
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



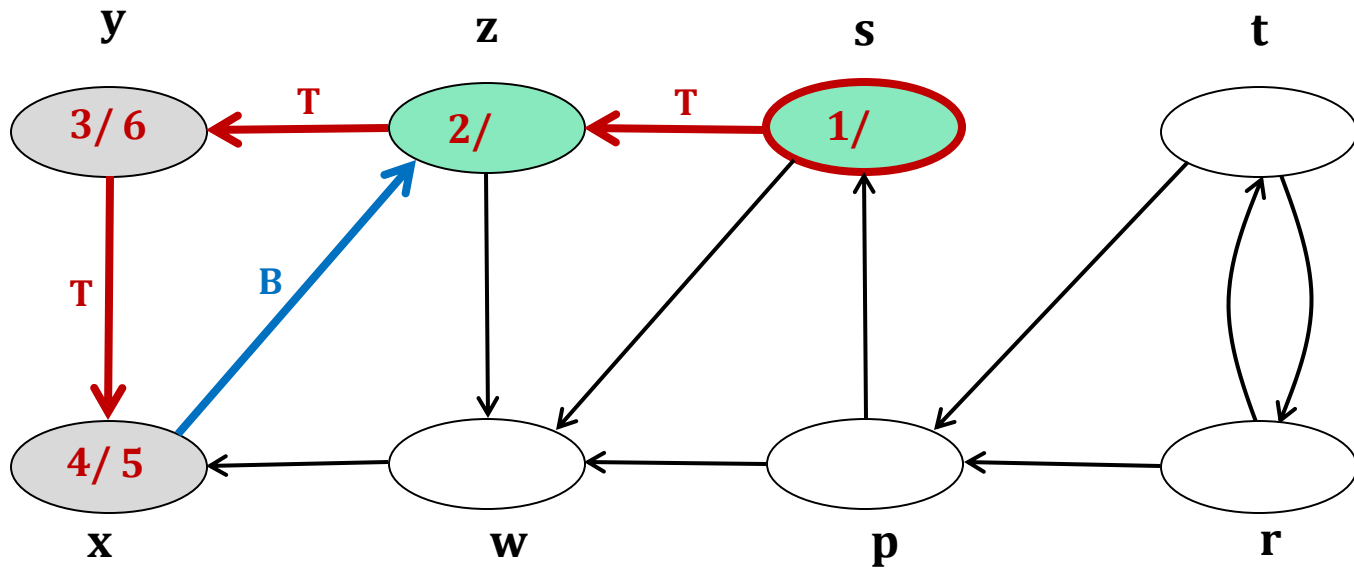
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



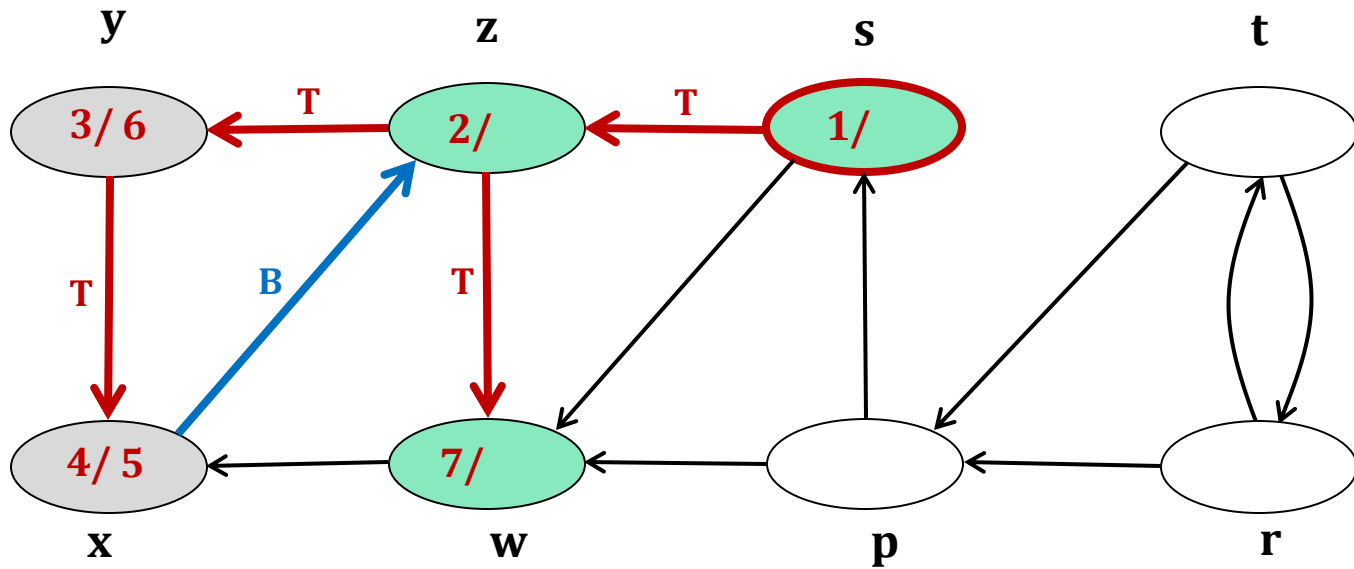
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



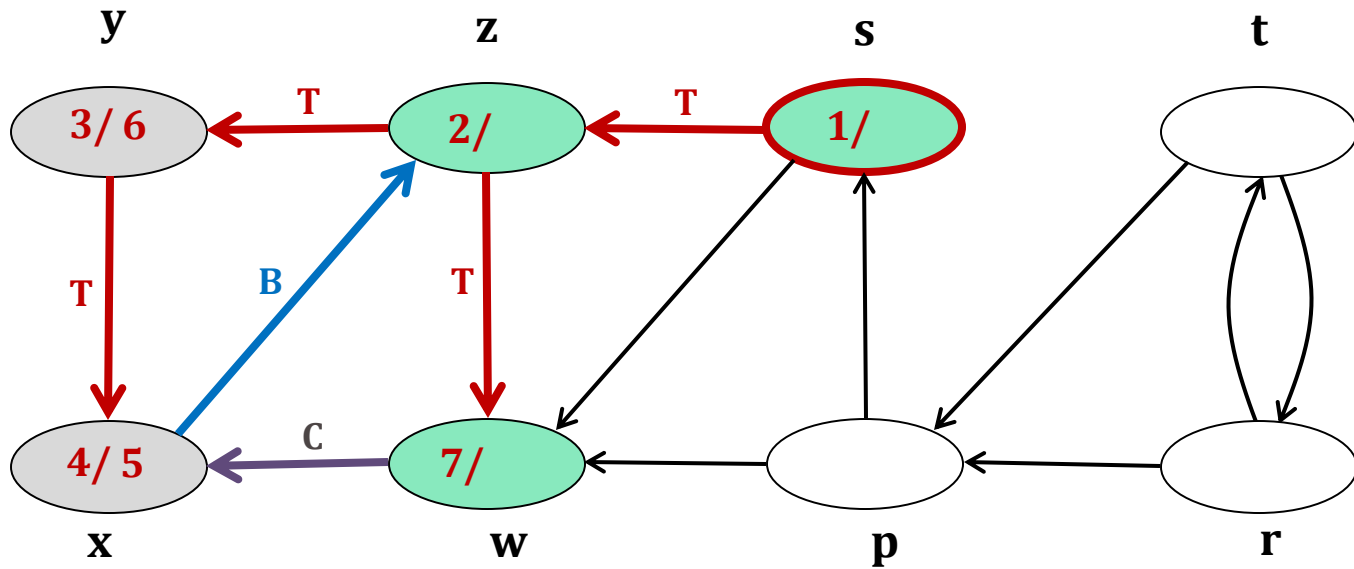
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



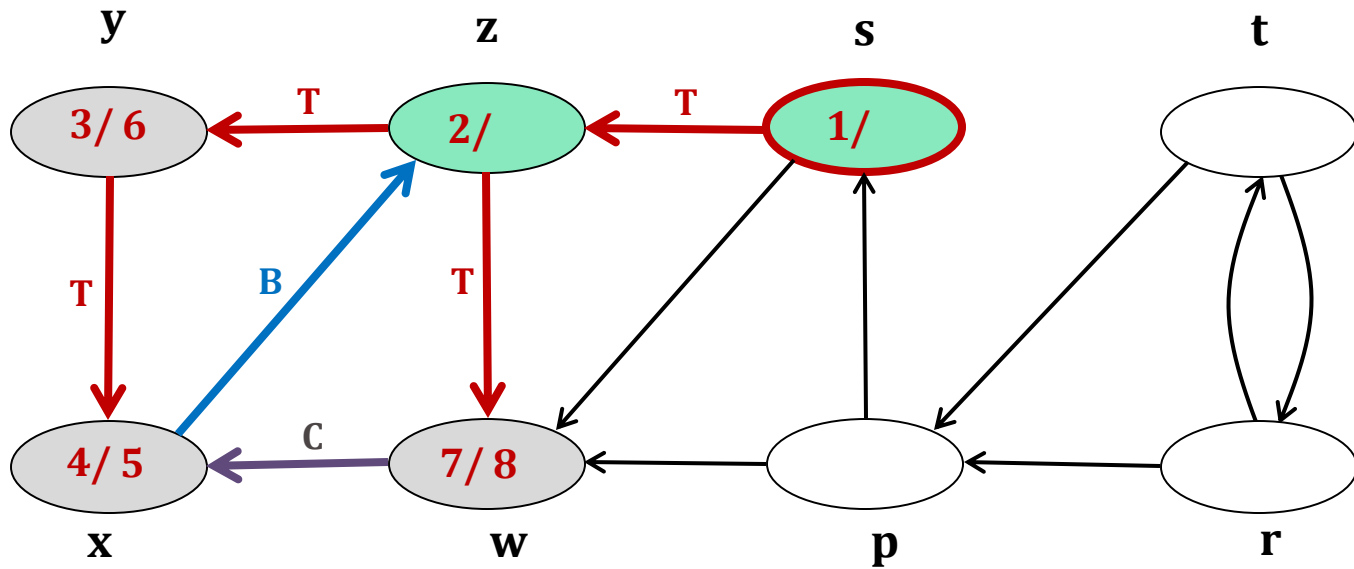
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



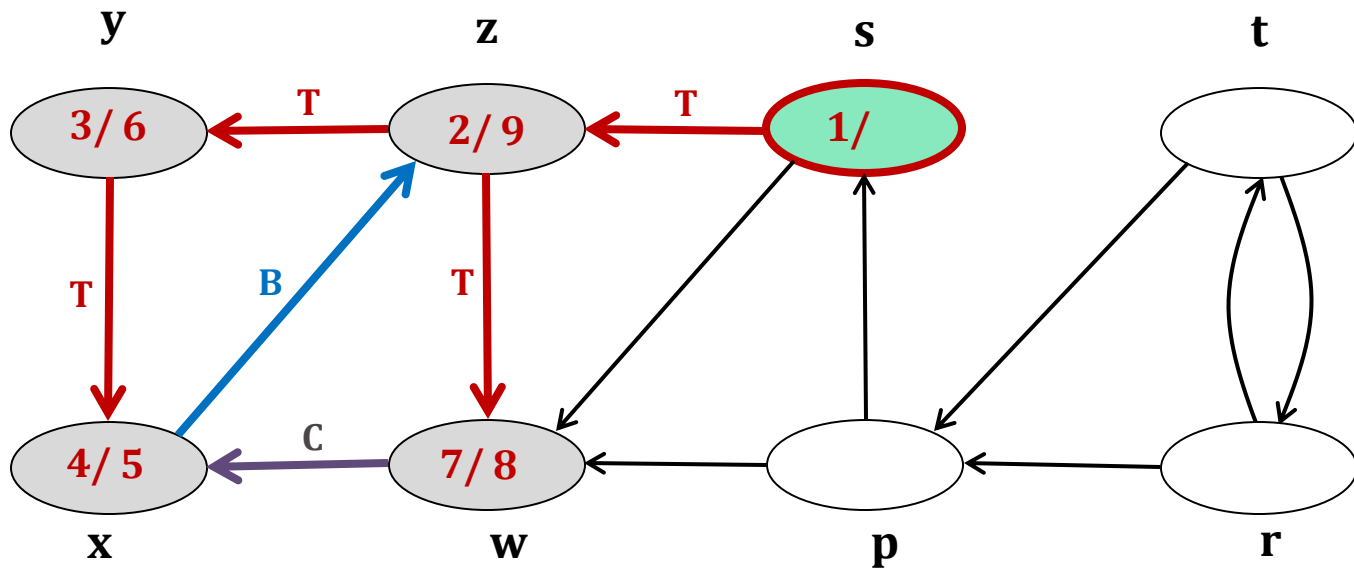
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



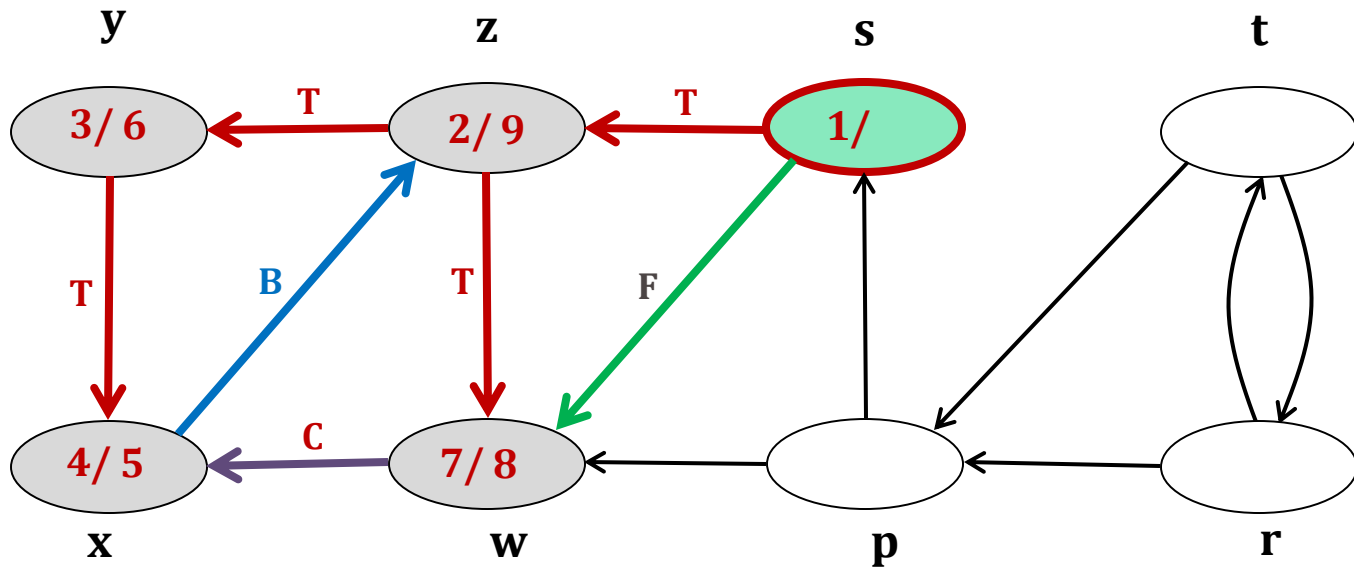
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



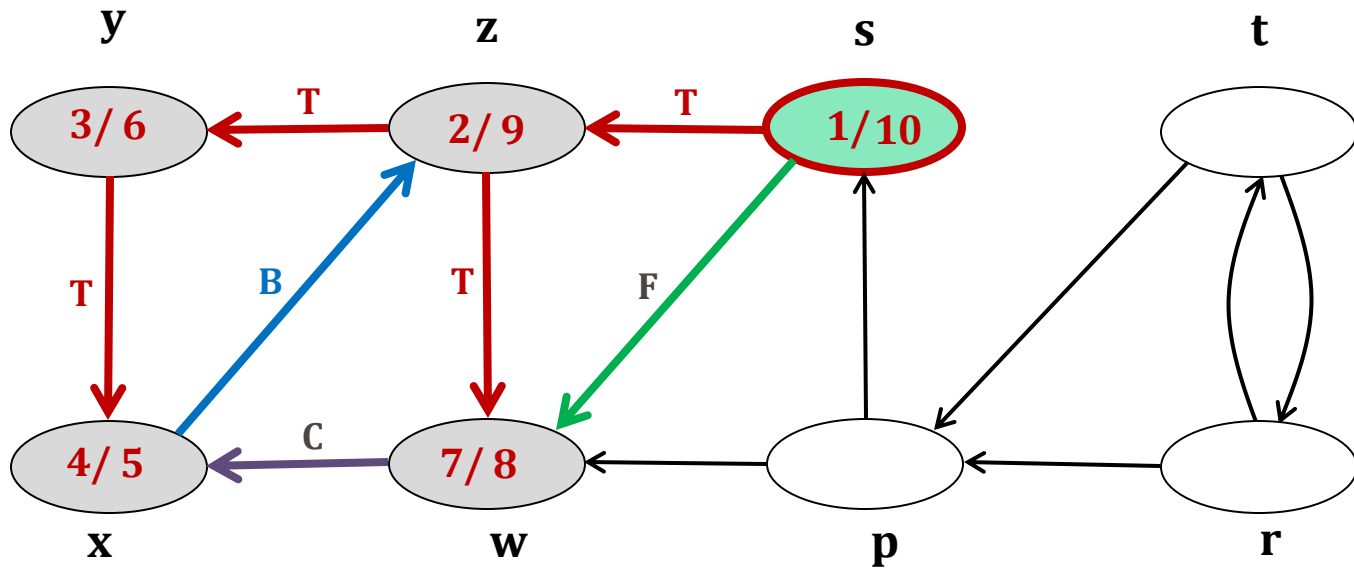
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



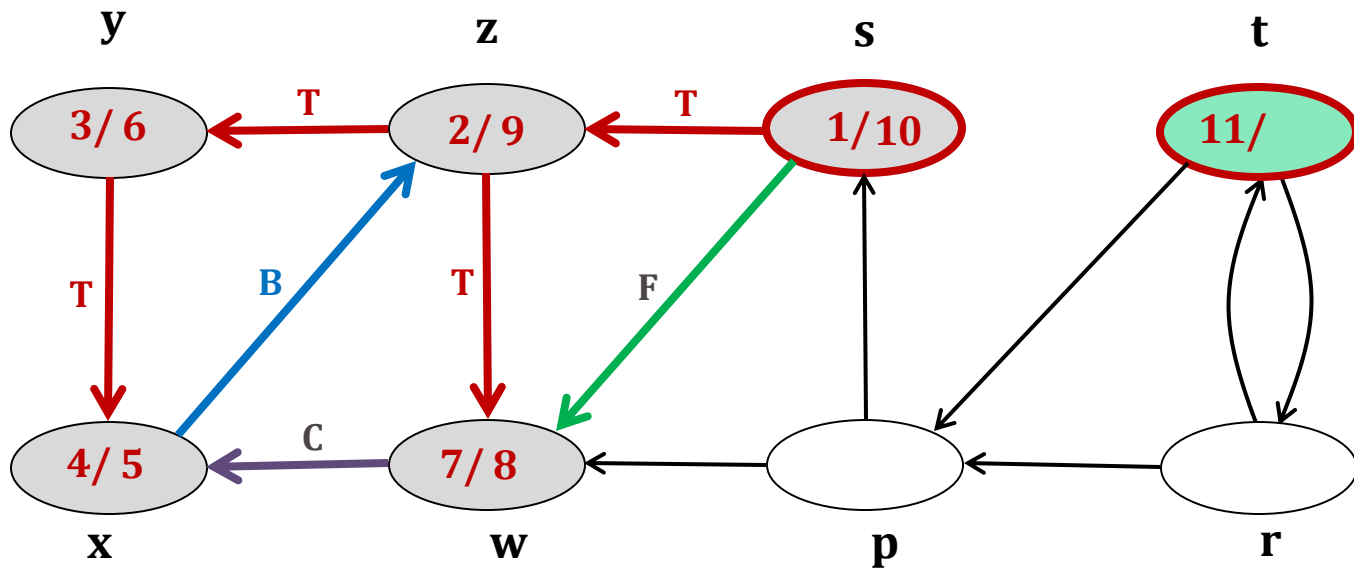
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



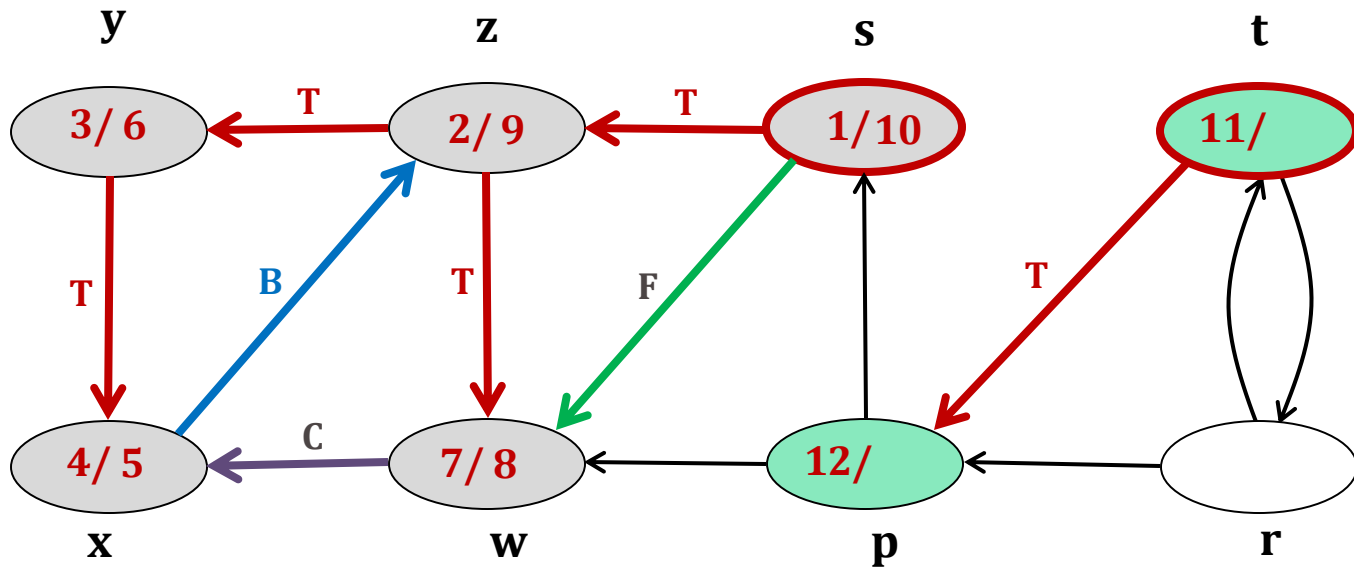
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



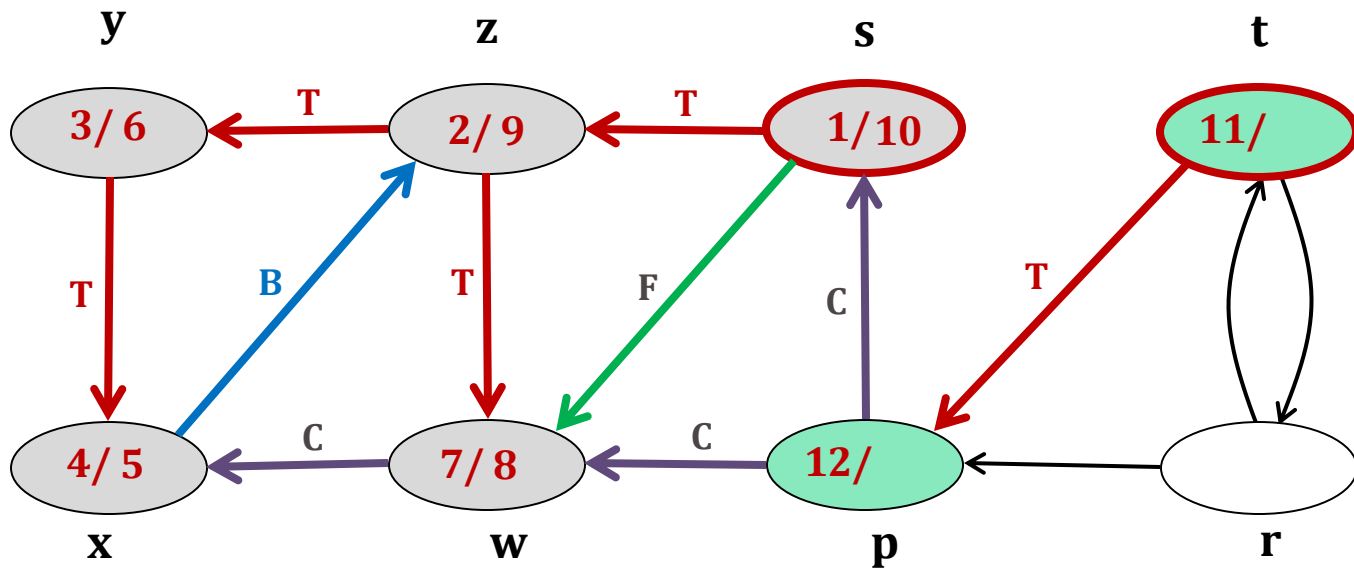
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



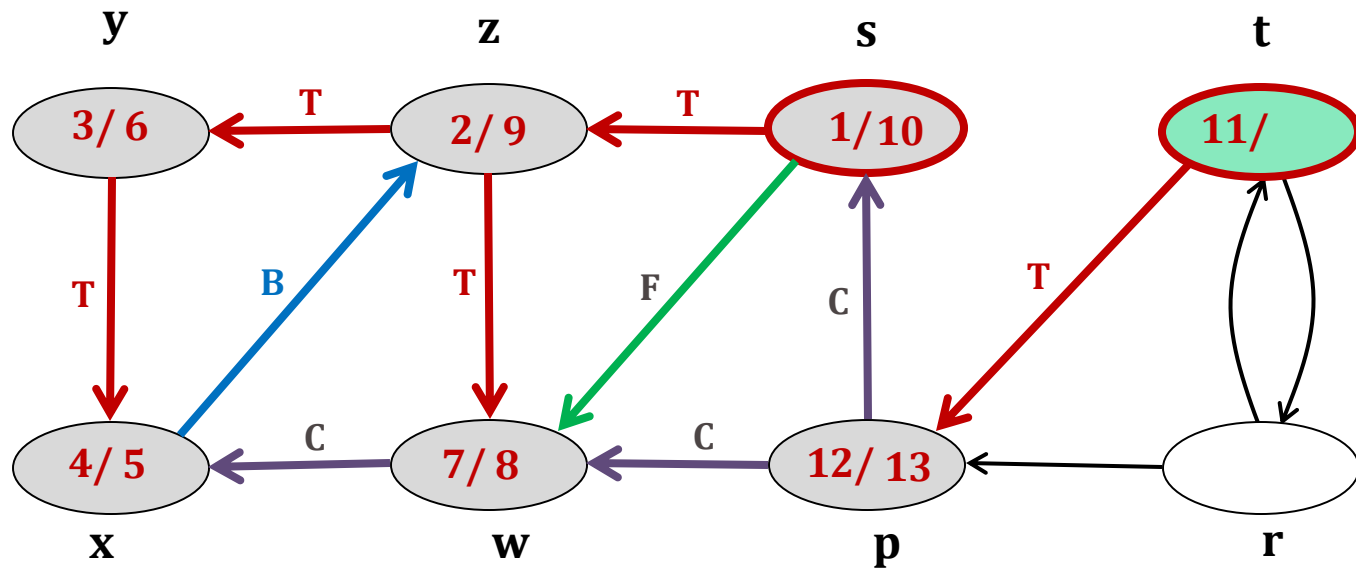
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



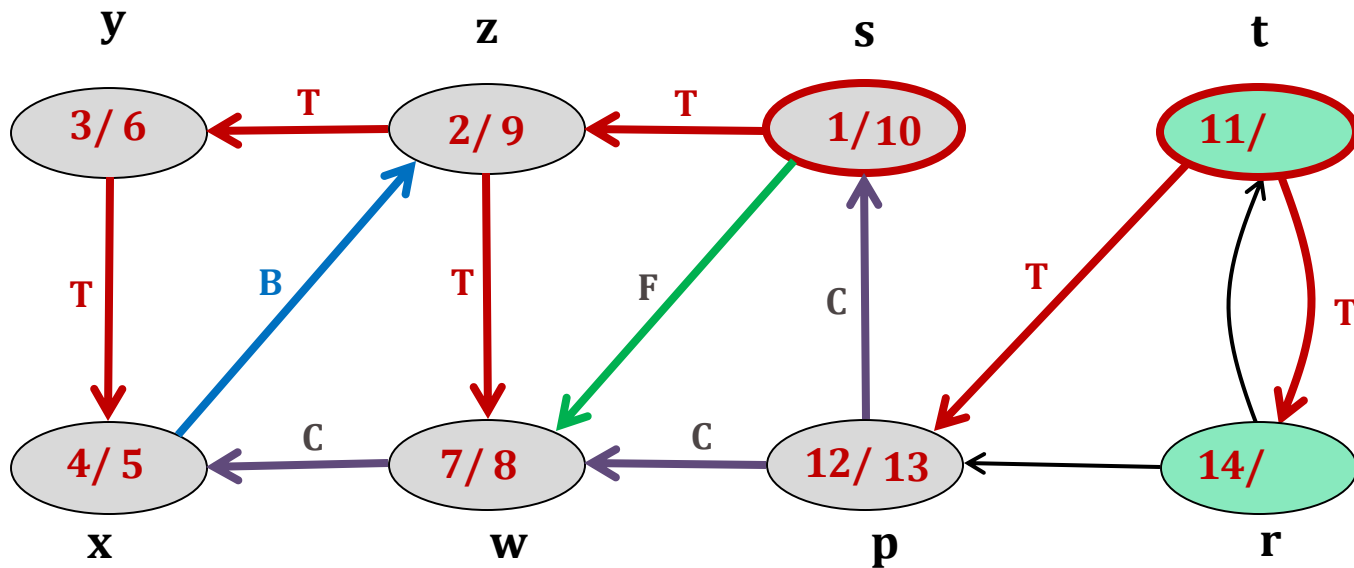
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



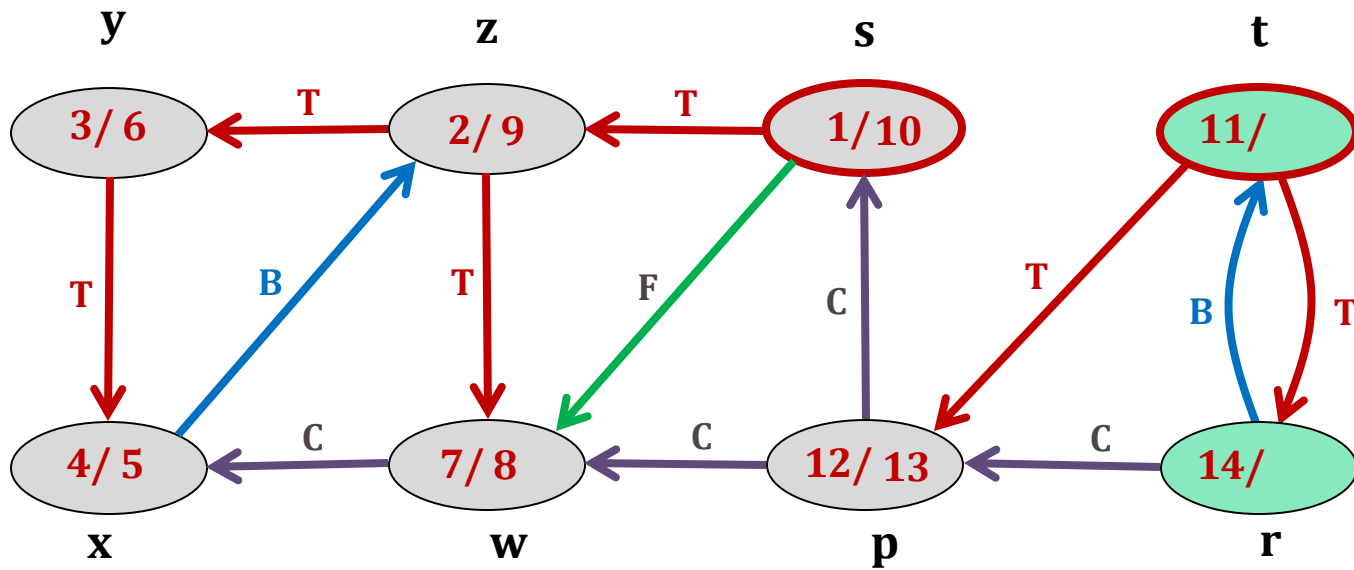
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



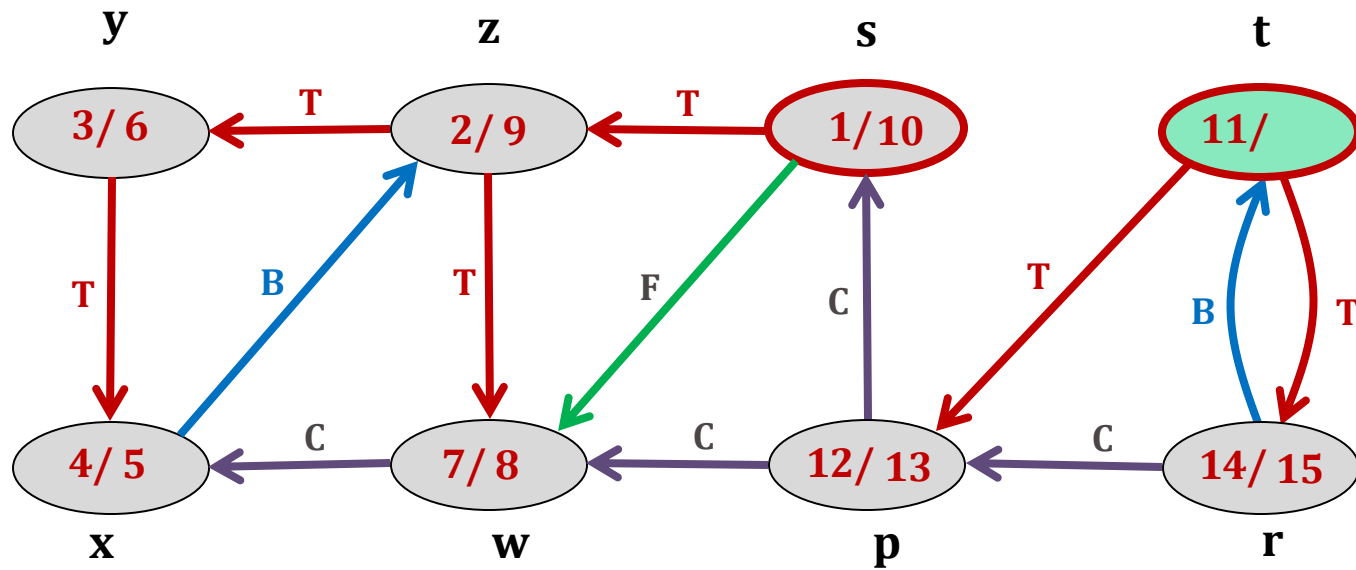
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



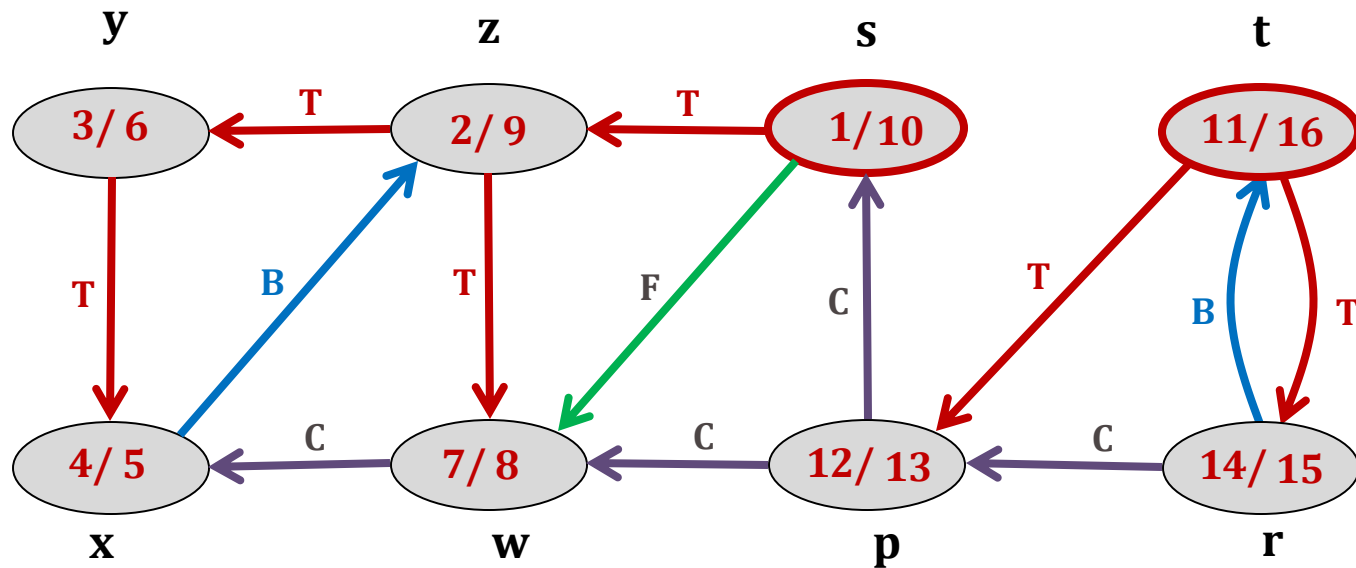
Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



Εξερεύνηση κατά Βάθος - DFS

Παράδειγμα



Τύπος ακμής (u,v)

T & **F**: $a(u) < a(v) < t(v) < t(u)$

B: $a(v) < a(u) < t(u) < t(v)$

C: $a(v) < t(v) < a(u) < t(u)$

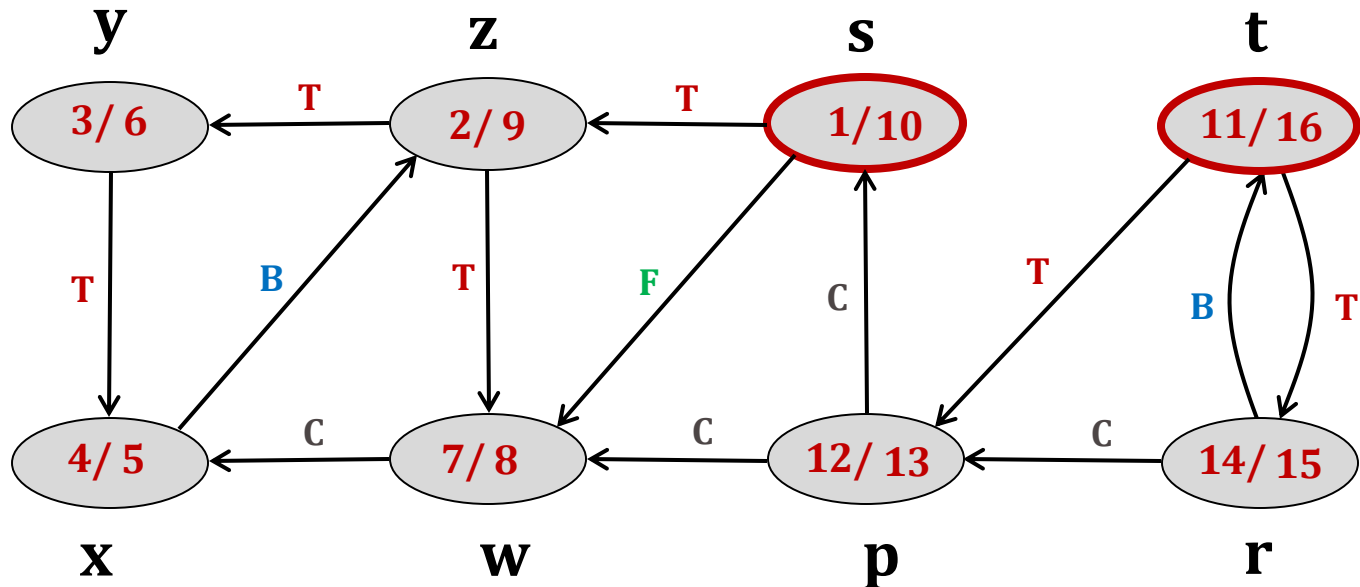
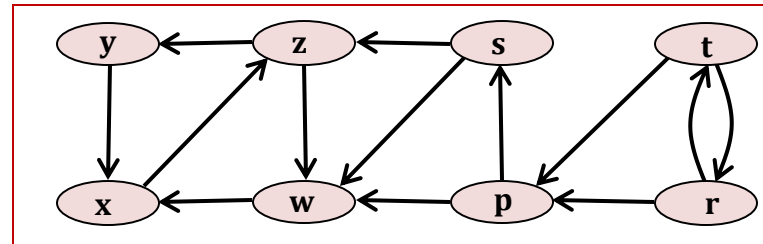
Μη κατευθ/νοι

γράφου:

μόνο **T** και **B**

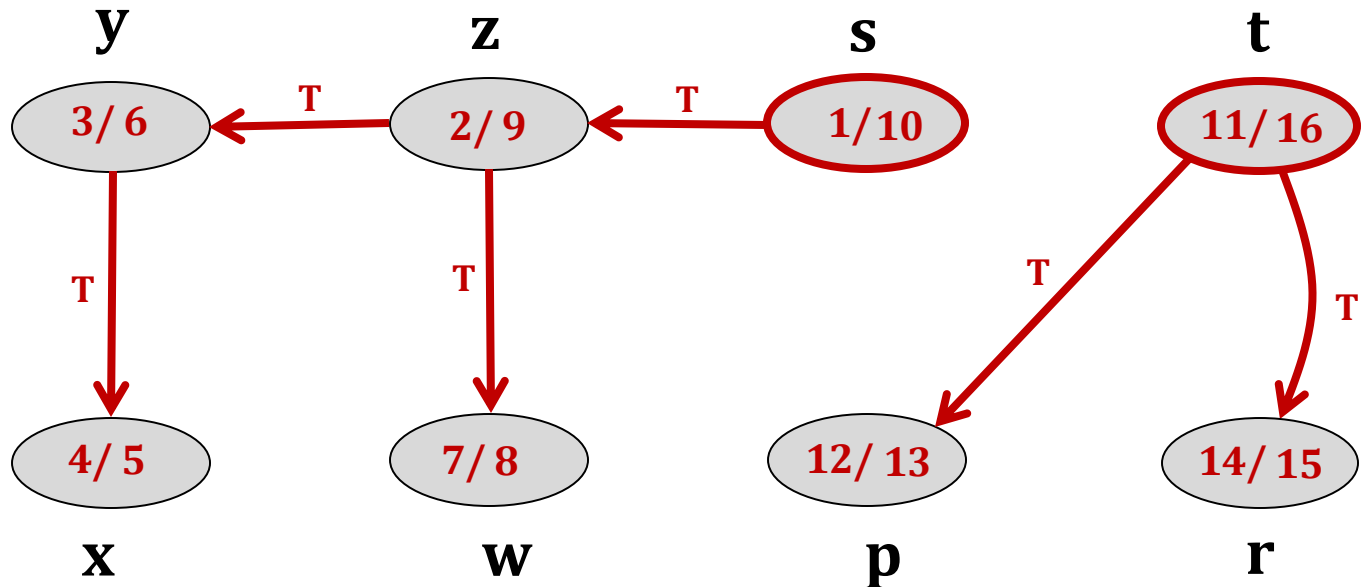
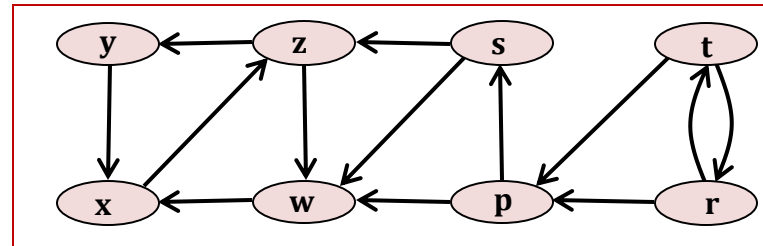
Εξερεύνηση κατά Βάθος - DFS

DFS-δένδρο (δάσος)



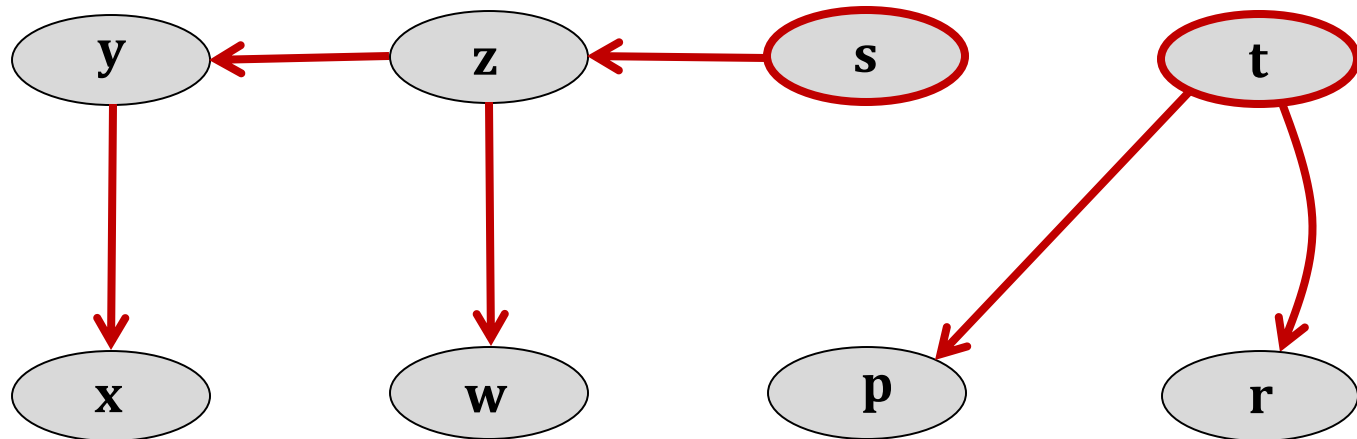
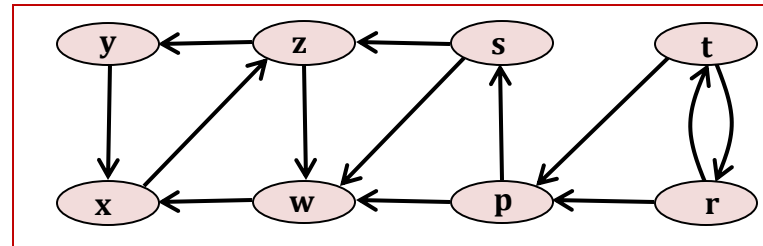
Εξερεύνηση κατά Βάθος - DFS

DFS-δένδρο (δάσος)



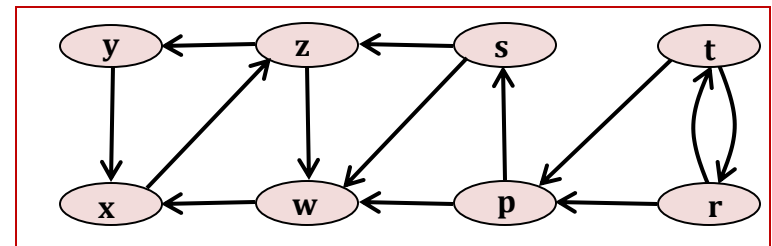
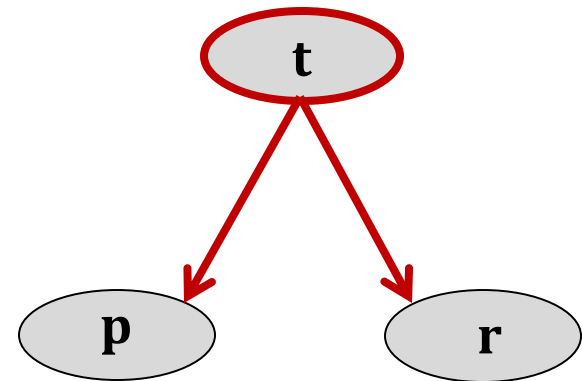
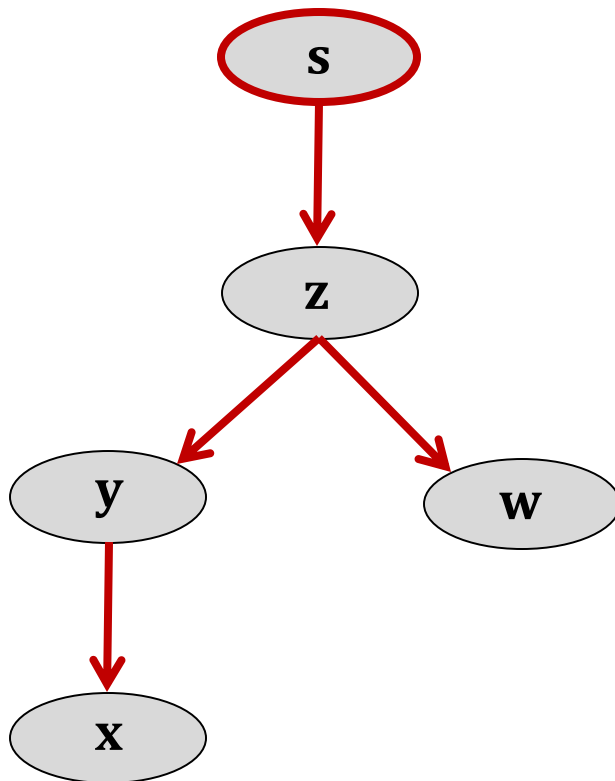
Εξερεύνηση κατά Βάθος - DFS

DFS-δένδρο (δάσος)



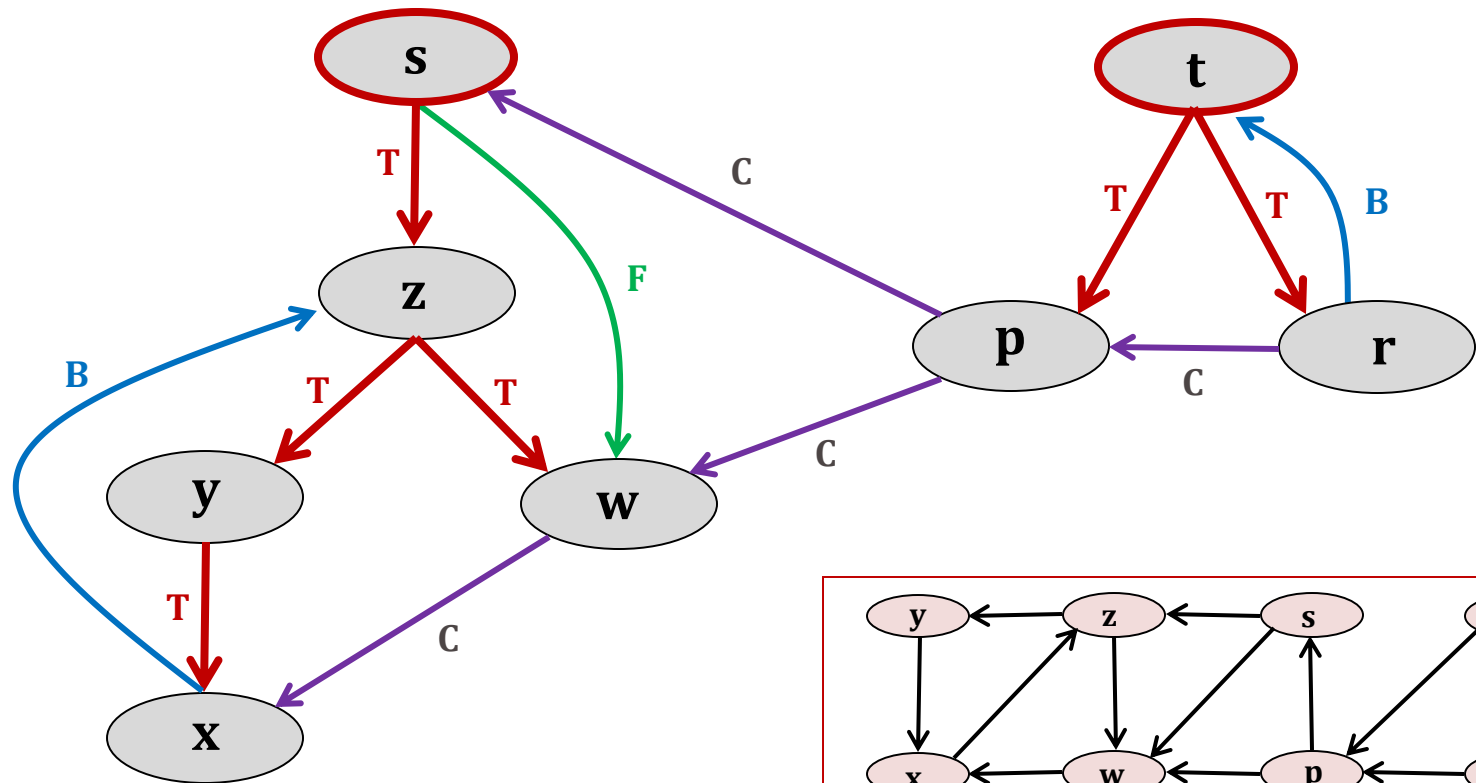
Εξερεύνηση κατά Βάθος - DFS

DFS-δένδρο (δάσος)

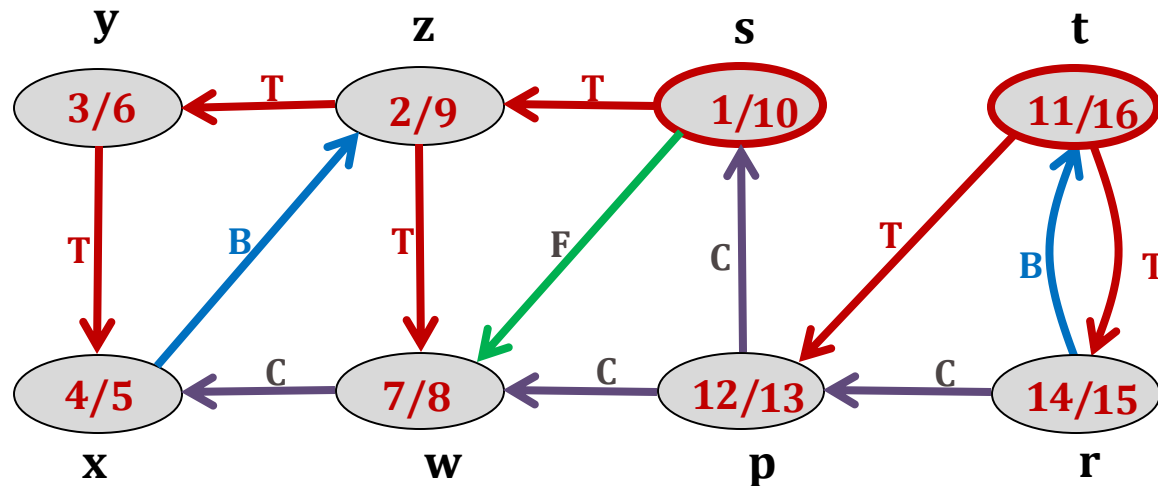


Εξερεύνηση κατά Βάθος - DFS

DFS-αναπαράσταση



Εφαρμογές DFS: εντοπισμός κύκλων



Τύπος ακμής (u,v)

T & F: $a(u) < a(v) < t(v) < t(u)$

B: $a(v) < a(u) < t(u) < t(v)$

C: $a(v) < t(v) < a(u) < t(u)$

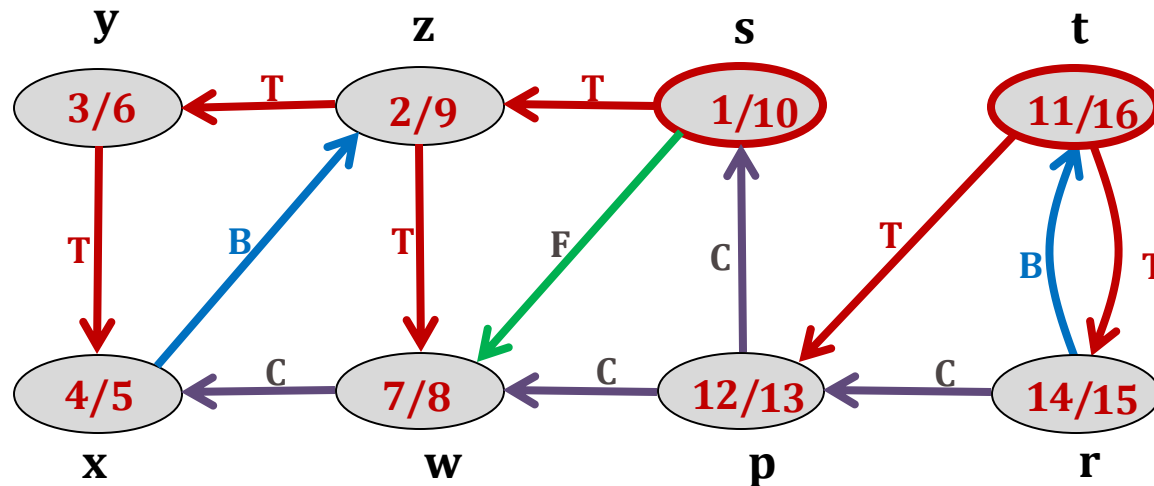
Μη κατευθ/νοι

γράφοι:

μόνο **T** και **B**

Εφαρμογές DFS: εντοπισμός κύκλων

- **Αναγκαία και ικανή συνθήκη** για ύπαρξη κύκλου σε γράφο $G=(V, E)$: η εξερεύνηση DFS βρίσκει **back edges**.



Τύπος ακμής (u,v)

T & F: $a(u) < a(v) < t(v) < t(u)$

B: $a(v) < a(u) < t(u) < t(v)$

C: $a(v) < t(v) < a(u) < t(u)$

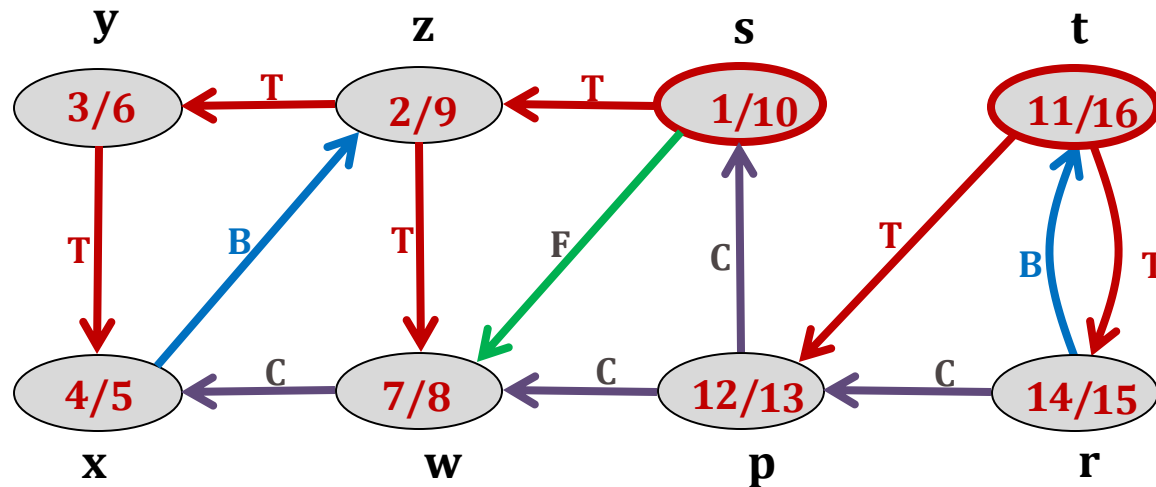
Μη κατευθ/νοι

γράφοι:

μόνο **T** και **B**

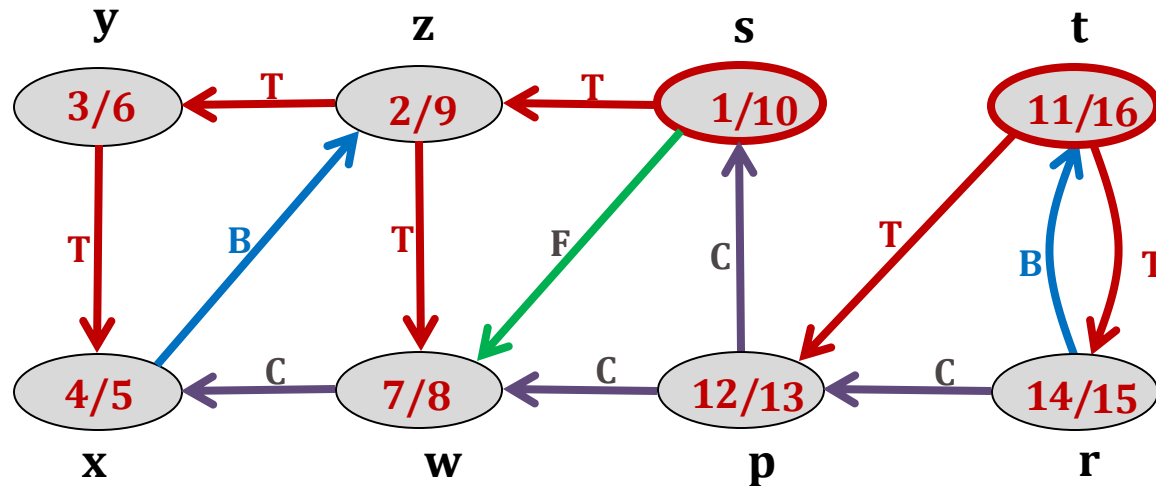
Εφαρμογές DFS: εντοπισμός κύκλων

- **Αναγκαία και ικανή συνθήκη** για ύπαρξη κύκλου σε γράφο $G=(V, E)$: η εξερεύνηση DFS βρίσκει **back edges**.



Εφαρμογές DFS: εντοπισμός κύκλων

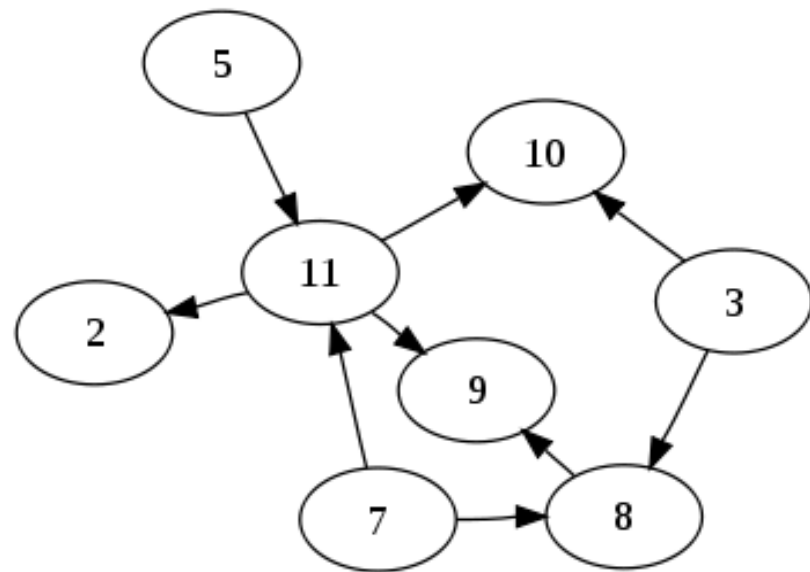
- **Αναγκαία και ικανή συνθήκη** για ύπαρξη κύκλου σε γράφο $G=(V, E)$: η εξερεύνηση DFS βρίσκει **back edges**.



- **Ικανή** : εύκολο!
- **Αναγκαία** : έστω κύκλος C και u ο πρώτος κόμβος του C που θα επισκεφθείη DFS. Σκεφτείτε τι θα κάνει η DFS κατά την εξερεύνηση του u.

Εφαρμογές DFS: Τοπολογική Ταξινόμηση

- Μη-κυκλικό Κατευθυνόμενο Γράφημα (DAG) ονομάζεται το γράφημα $G=(V, E)$ το οποίο δεν έχει κατευθυνόμενους κύκλους.



Εφαρμογές DFS: Τοπολογική Ταξινόμηση

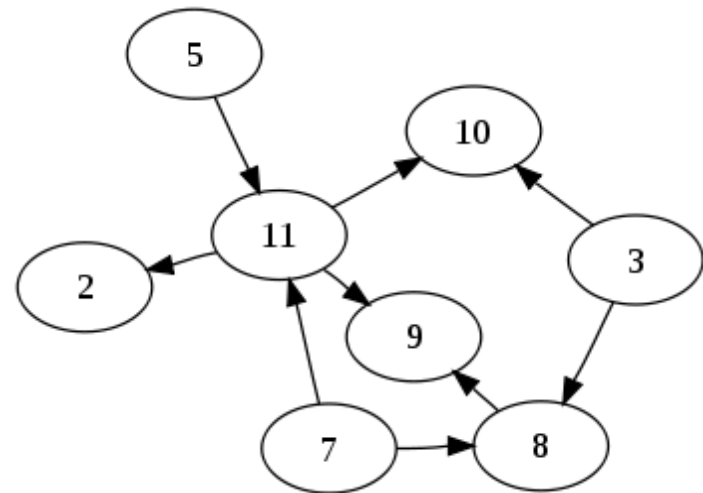


Μια **Τοπολογική Ταξινόμηση** ενός DAG $G=(V, E)$ είναι μια ακολουθία $TS=(u_1, u_2, \dots, u_n)$ των κόμβων του $V(G)$ τέτοια ώστε:

$\forall u_i, u_j \in V(G)$ εάν $\langle u_i, u_j \rangle \in E(G)$ τότε ο κόμβος u_i προηγείται του u_j στην ακολουθία **TS**

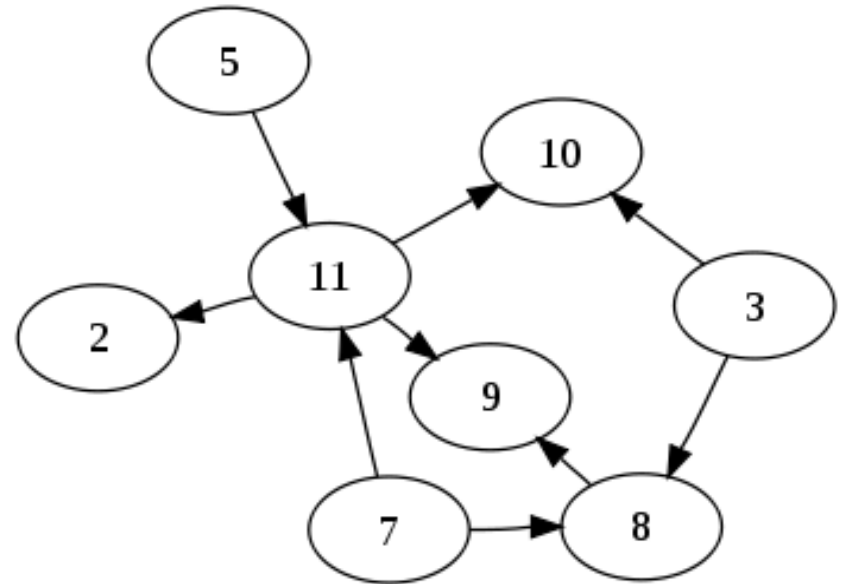
Υπολογισμός TS

- ✓ Μέθοδος **In-degree**
- ✓ Μέθοδος **DFS**



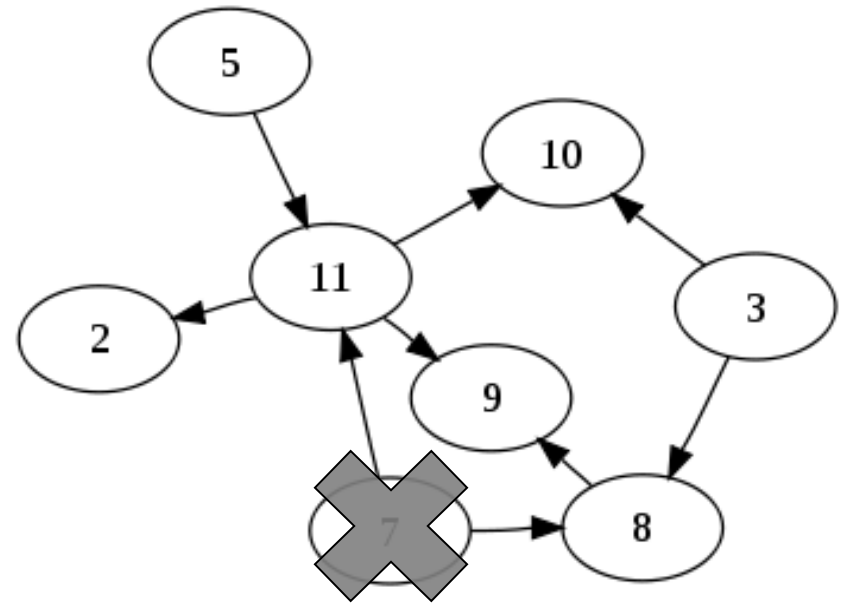
Εφαρμογές DFS: Τοπολογική Ταξινόμηση

■ Μέθοδος **In-degree**



Εφαρμογές DFS: Τοπολογική Ταξινόμηση

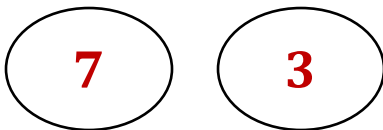
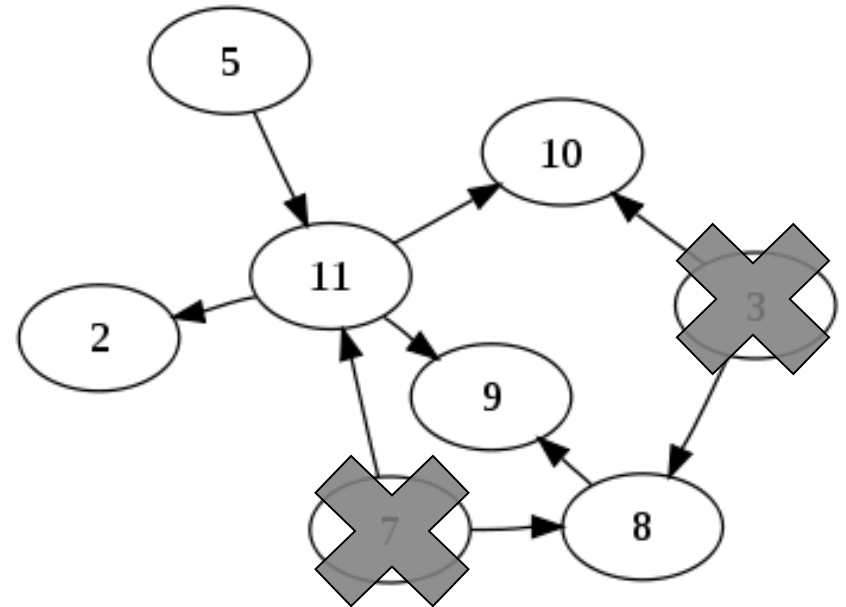
■ Μέθοδος **In-degree**



7

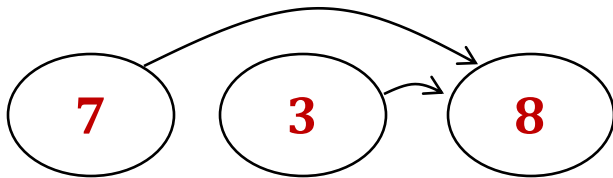
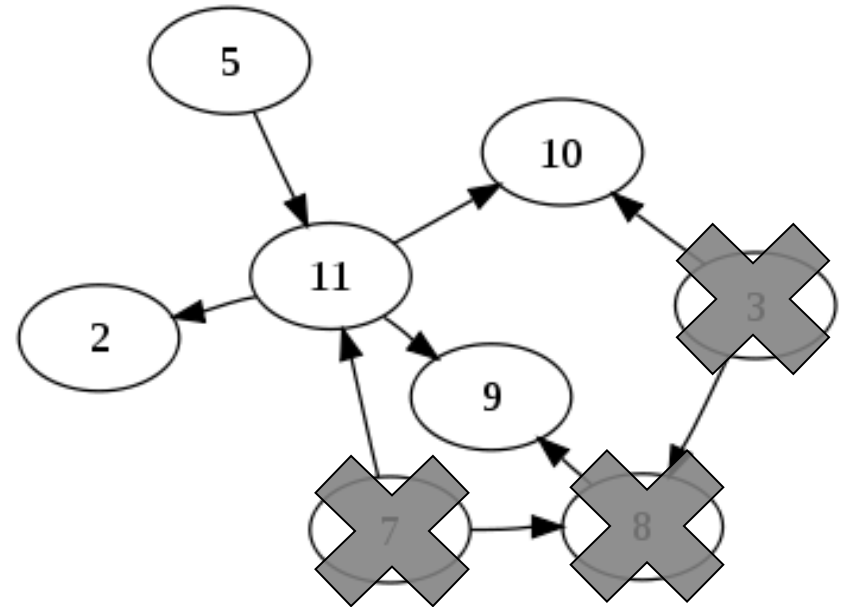
Εφαρμογές DFS: Τοπολογική Ταξινόμηση

■ Μέθοδος **In-degree**



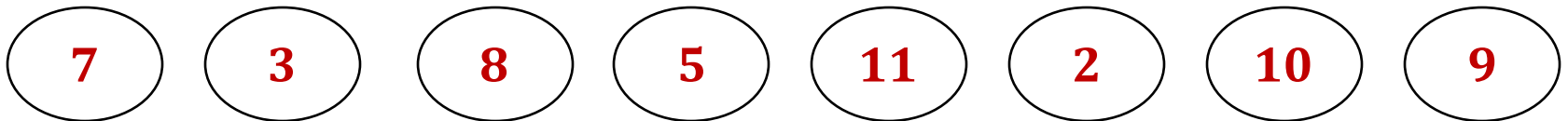
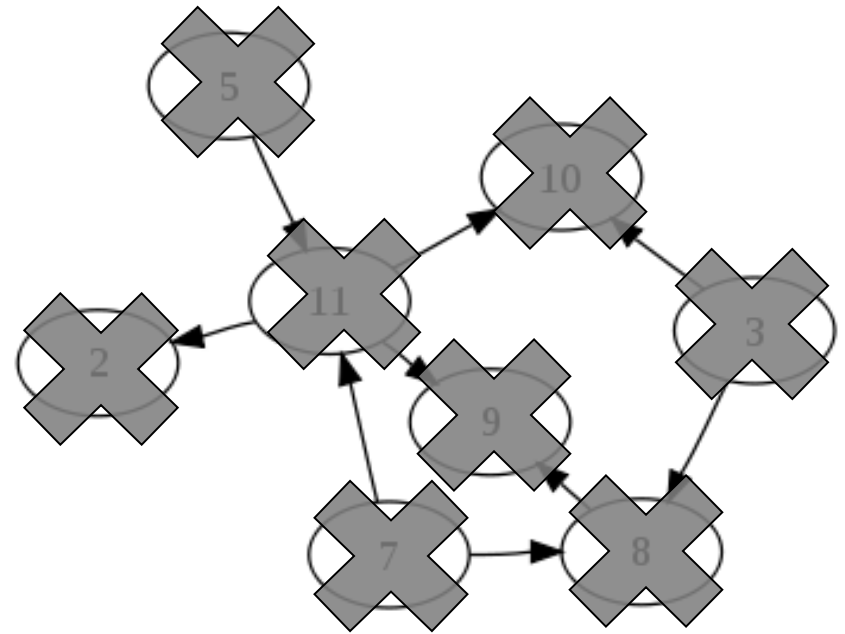
Εφαρμογές DFS: Τοπολογική Ταξινόμηση

■ Μέθοδος **In-degree**



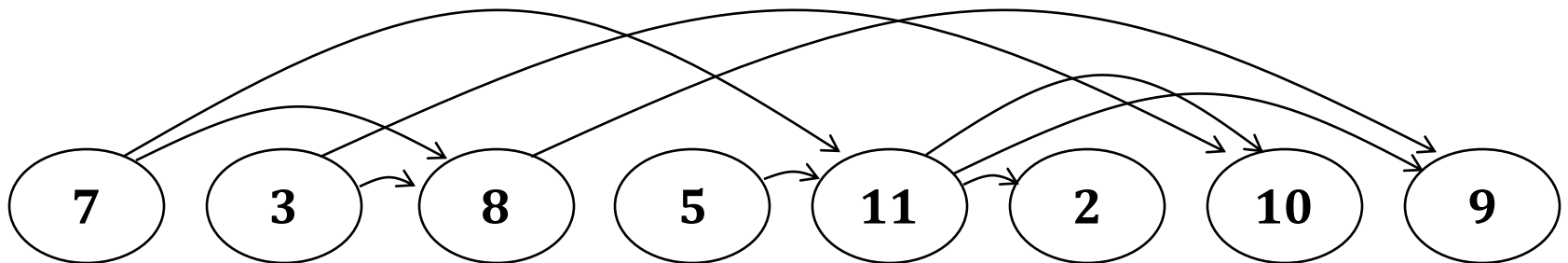
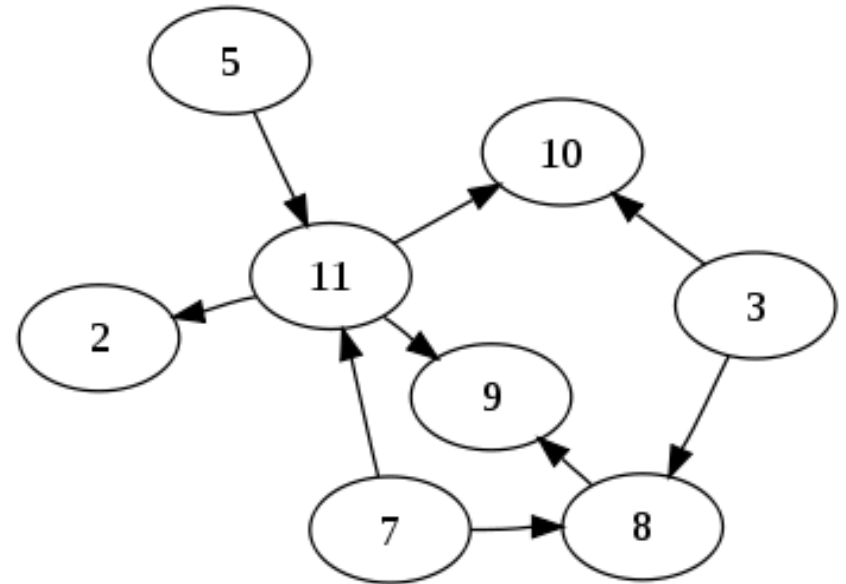
Εφαρμογές DFS: Τοπολογική Ταξινόμηση

■ Μέθοδος **In-degree**



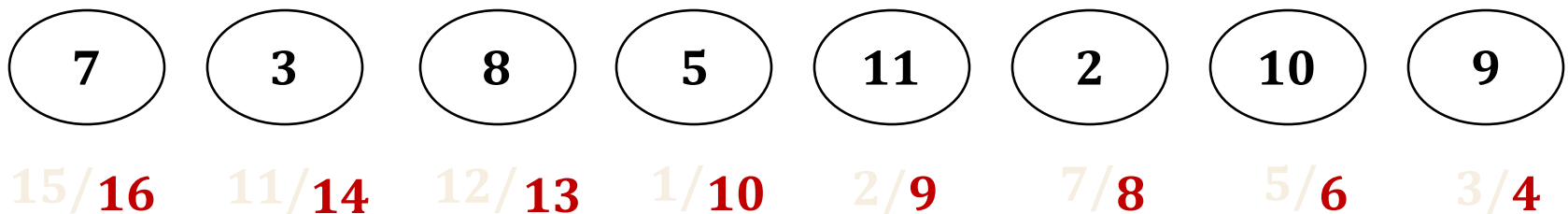
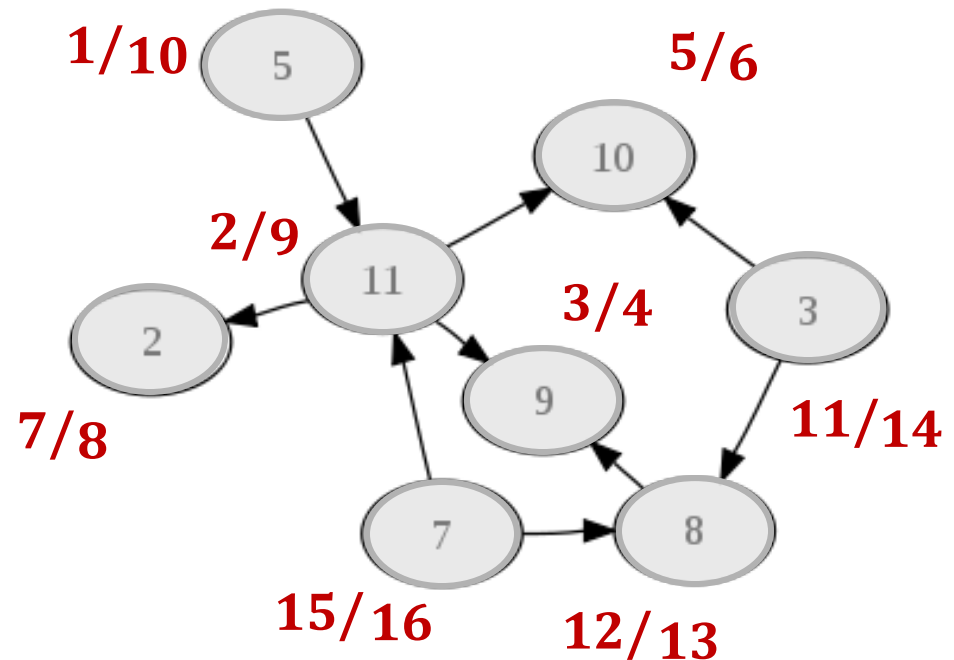
Εφαρμογές DFS: Τοπολογική Ταξινόμηση

■ Μέθοδος **In-degree**



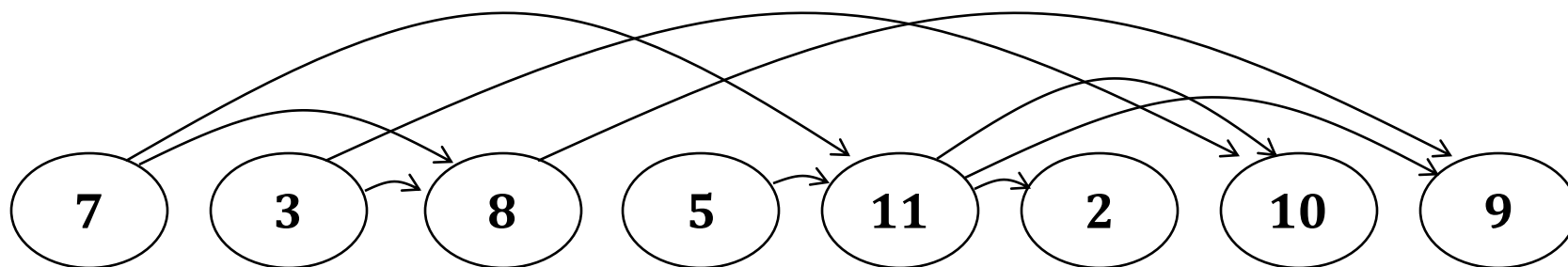
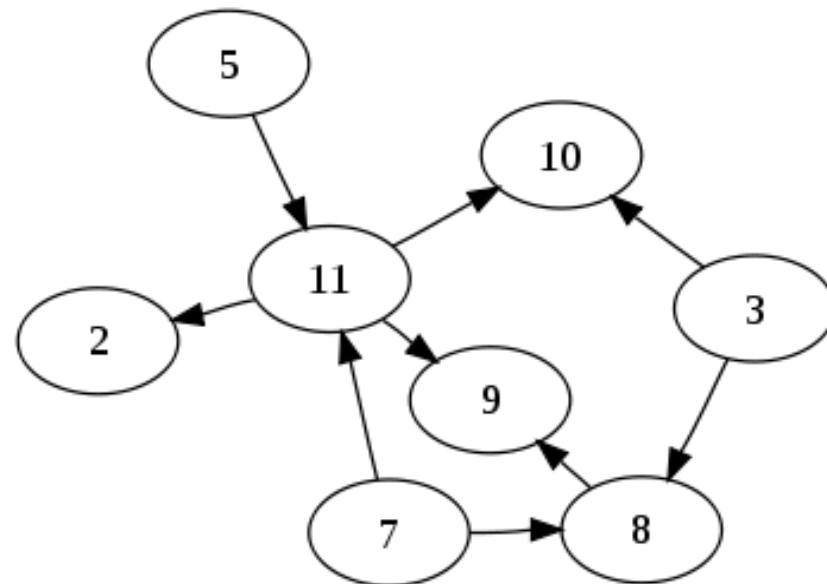
Εφαρμογές DFS: Τοπολογική Ταξινόμηση

■ Μέθοδος **DFS**



Εφαρμογές DFS: Τοπολογική Ταξινόμηση

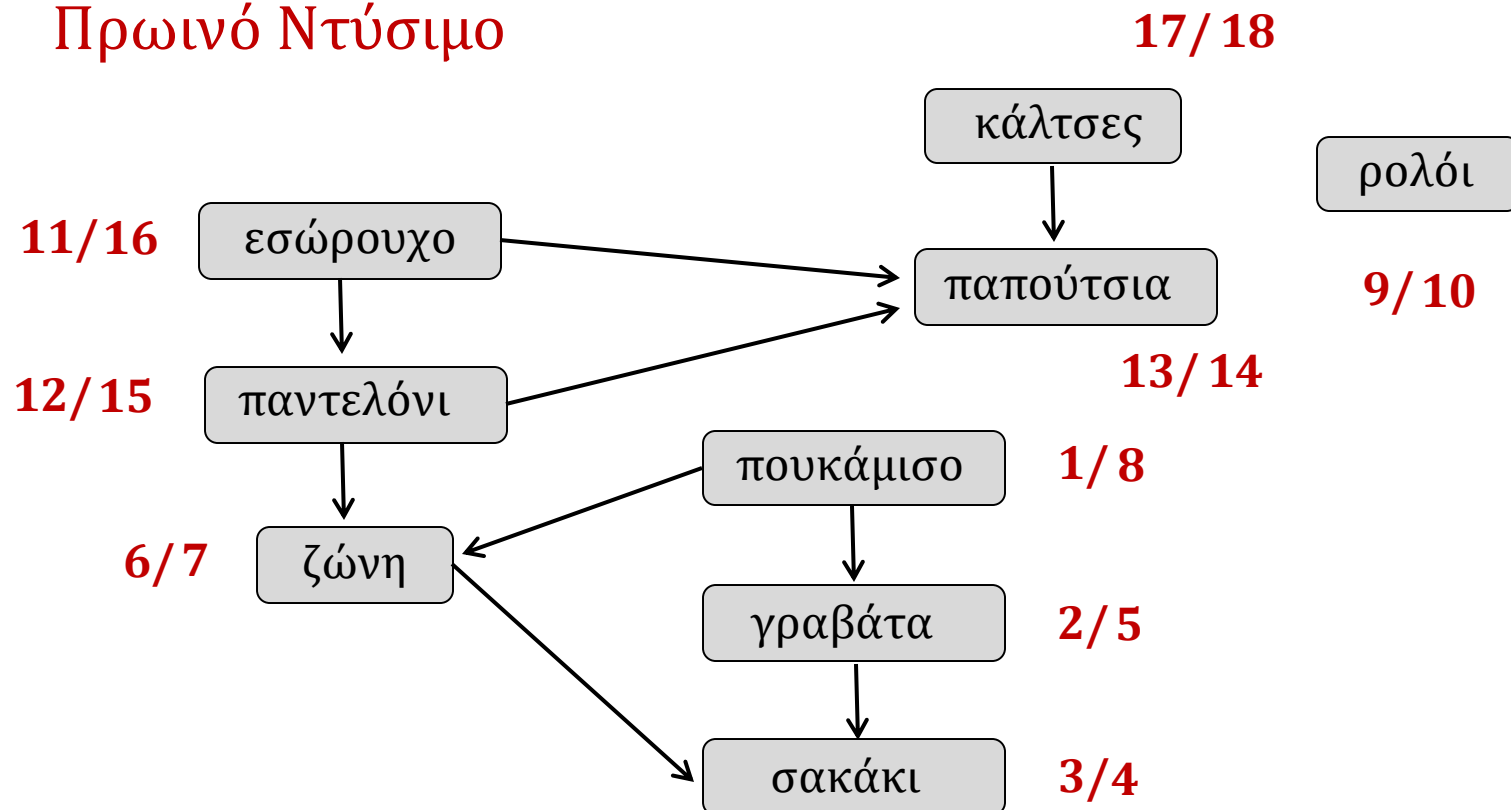
■ Μέθοδος **DFS**



Εφαρμογές Τοπολογικής Ταξινόμησης: scheduling

Ε

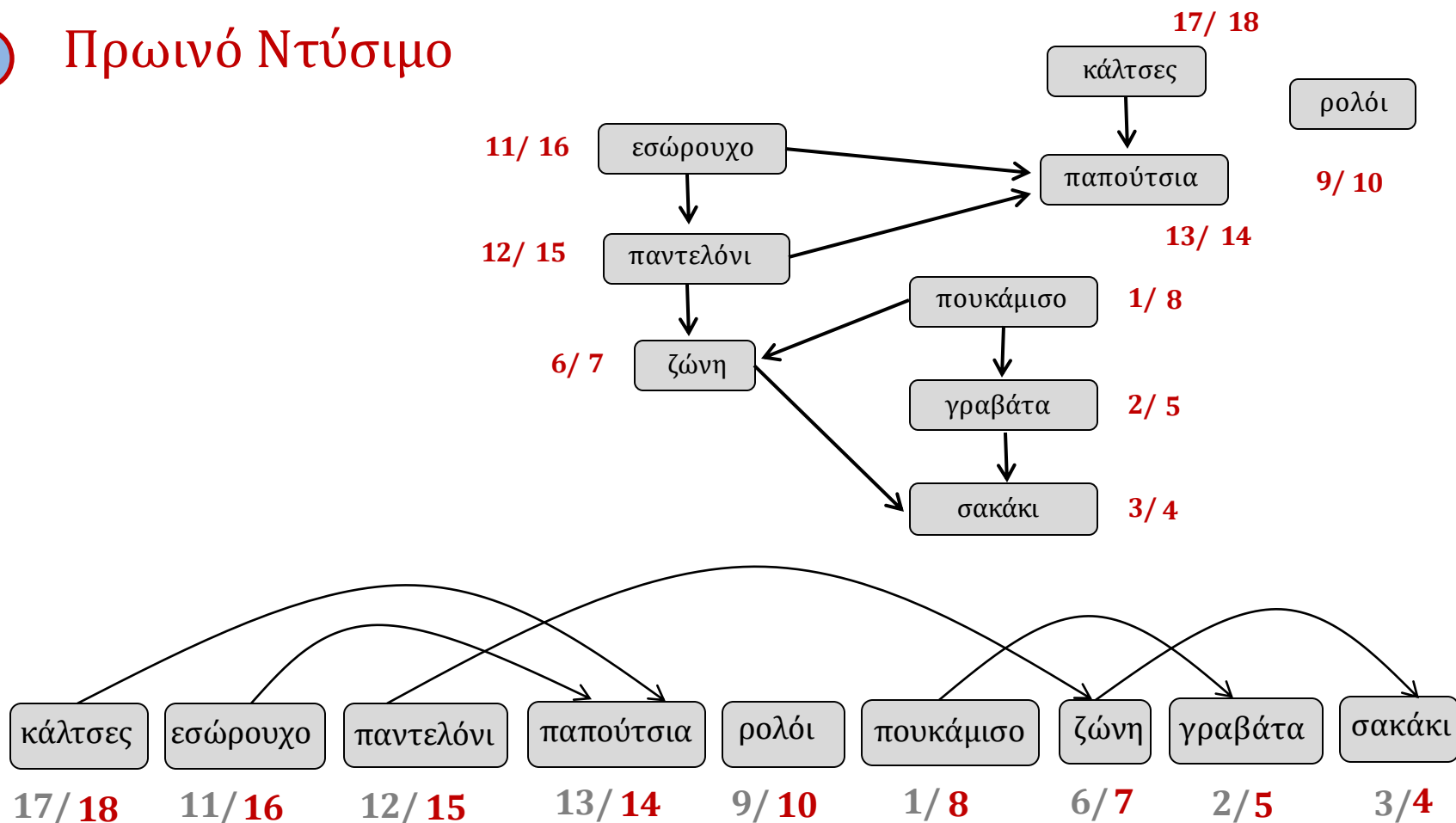
Πρωινό Ντύσιμο



Εφαρμογές TS

Ε

Πρωινό Ντύσιμο



Και λίγο παιχνίδι!

● Το Πρόβλημα του Βαρκάρη

Ένας βαρκάρης θέλει να περάσει στην
απέναντι όχθη ενός ποταμού
έναν **Λύκο**, ένα **Πρόβατο** και ένα
Λάχανο ή **Χορτάρι** !



Πως είναι δυνατόν ο βοσκός να τα
περάσει απέναντι, έτσι ώστε να μην
φάει το **πρόβατο** το **χορτάρι**, ή
ο **λύκος** το **πρόβατο** ;

Το Πρόβλημα του Βαρκάρη

■ Ο Λύκος, το Πρόβατο και το Χορτάρι



Το Πρόβλημα του Βαρκάρη: μετατροπή σε εξερεύνηση

■ Επίλυση με γράφο (διάγραμμα) καταστάσεων

1	ΛΠΧ - Β	
2	ΠΧ - Β	Λ
3	ΛΧ - Β	Π
4	ΛΠ - Β	Χ
5	Π - Β	ΛΧ
6	ΛΧ	Β - Π
7	Χ	Β - ΛΠ
8	Π	Β - ΛΧ
9	Λ	Β - ΠΧ
10		Β - ΛΠΧ



Χ Π Λ - Β |

Αρχική
Κατάσταση



| Β - Λ Π Χ

Τελική
Κατάσταση

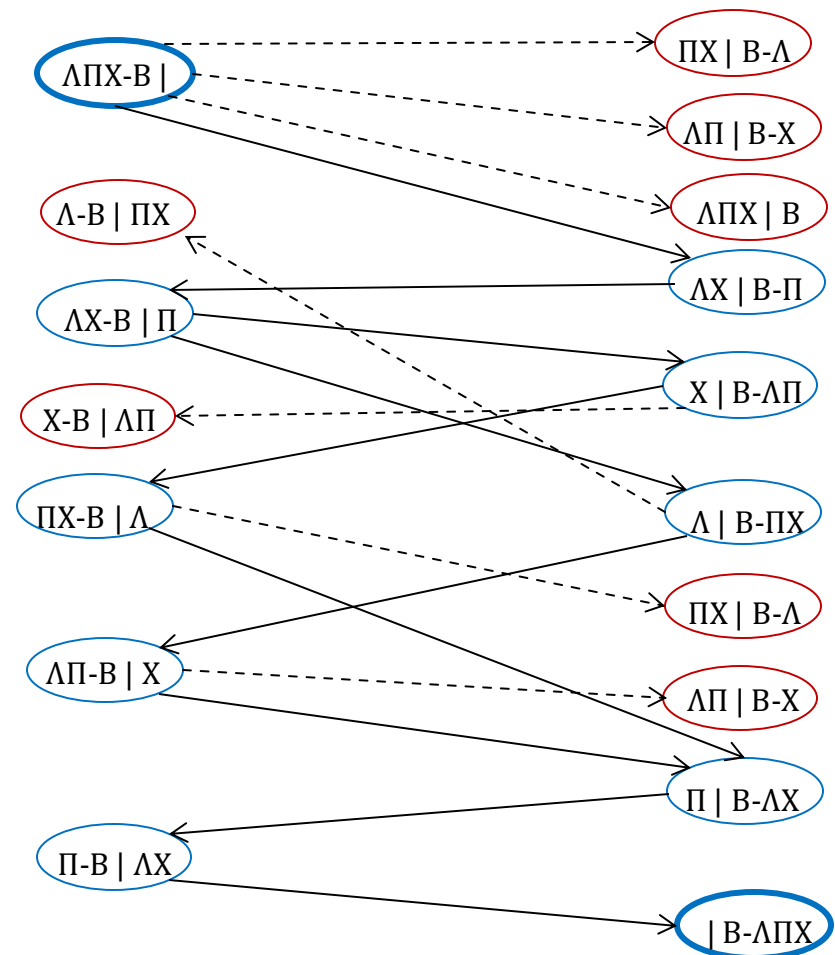
ΕΠΙΤΡΕΠΤΕΣ ΚΑΤΑΣΤΑΣΕΙΣ

Το Πρόβλημα του Βαρκάρη: μετατροπή σε εξερεύνηση

■ **Πρόβλημα Βαρκάρη = Εύρεση διαδρομής σε γράφο**

1	ΛΠΧ-Β	
2	ΠΧ-Β	Λ
3	ΛΧ-Β	Π
4	ΛΠ-Β	Χ
5	Π-Β	ΛΧ
6	ΛΧ	Β-Π
7	Χ	Β-ΛΠ
8	Π	Β-ΛΧ
9	Λ	Β-ΠΧ
10		Β-ΛΠΧ

ΕΠΙΤΡΕΠΤΕΣ ΚΑΤΑΣΤΑΣΕΙΣ

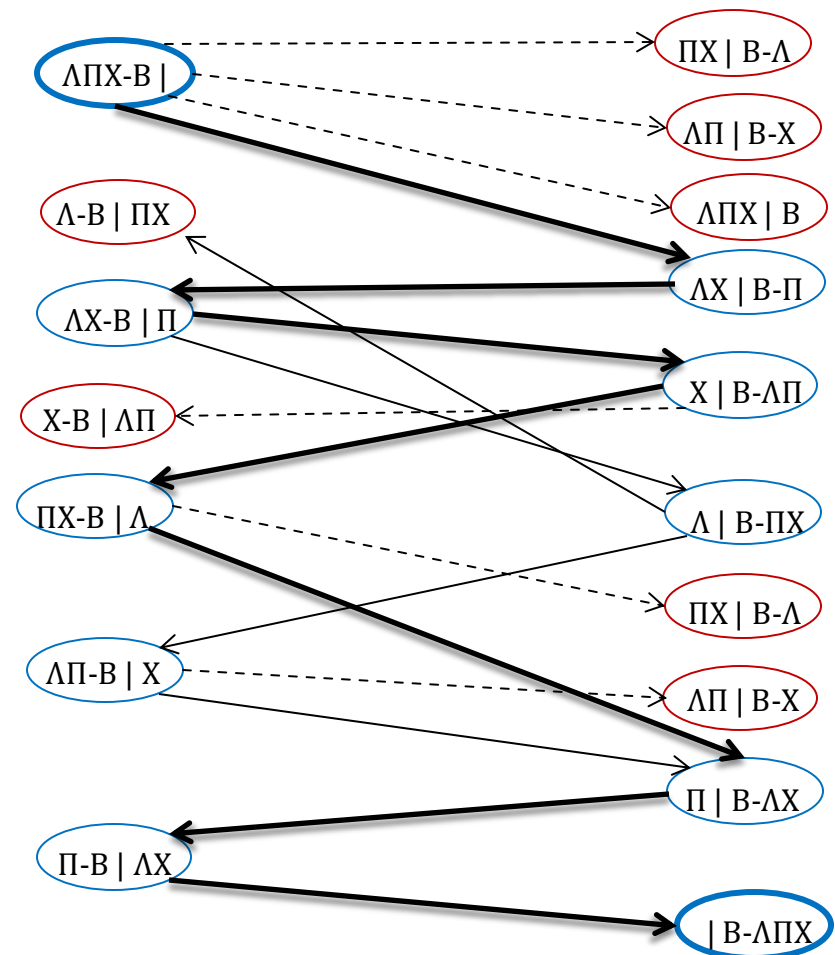


Το Πρόβλημα του Βαρκάρη : μετατροπή σε εξερεύνηση

■ **Πρόβλημα Βαρκάρη = Εύρεση διαδρομής σε γράφο**

1	ΛΠΧ-Β	
2	ΠΧ-Β	Λ
3	ΛΧ-Β	Π
4	ΛΠ-Β	Χ
5	Π-Β	ΛΧ
6	ΛΧ	Β-Π
7	Χ	Β-ΛΠ
8	Π	Β-ΛΧ
9	Λ	Β-ΠΧ
10		Β-ΛΠΧ

ΕΠΙΤΡΕΠΤΕΣ ΚΑΤΑΣΤΑΣΕΙΣ

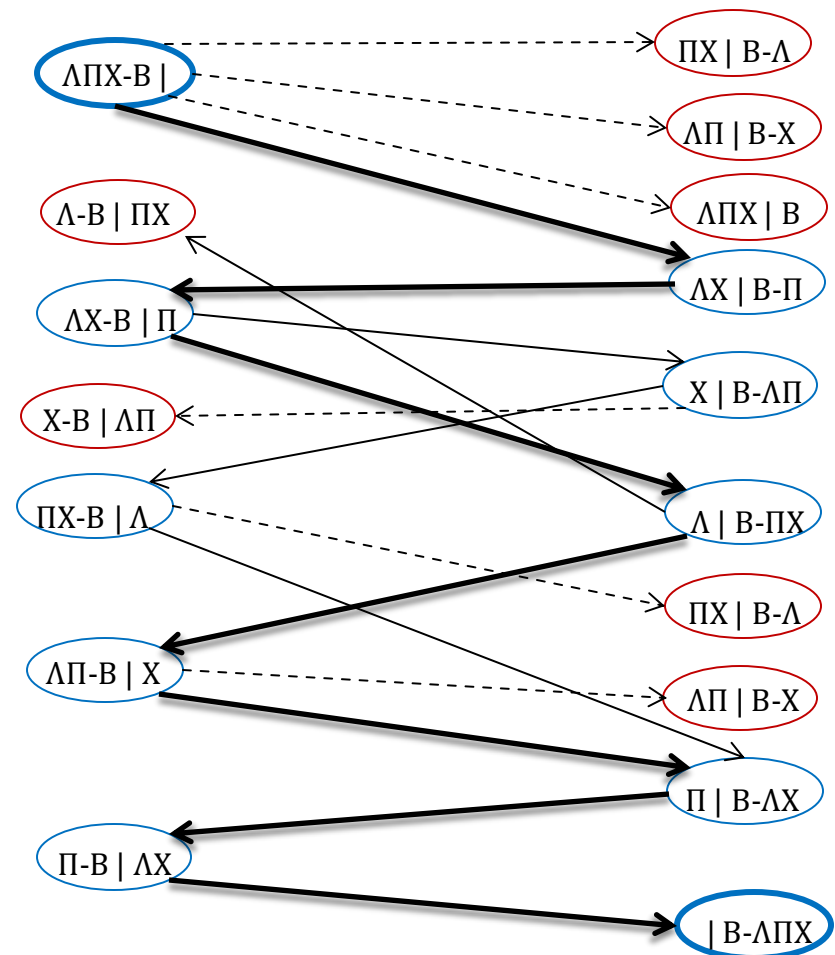


Το Πρόβλημα του Βαρκάρη : μετατροπή σε εξερεύνηση

■ **Πρόβλημα Βαρκάρη = Εύρεση διαδρομής σε γράφο**

1	ΛΠΧ-Β	
2	ΠΧ-Β	Λ
3	ΛΧ-Β	Π
4	ΛΠ-Β	Χ
5	Π-Β	ΛΧ
6	ΛΧ	Β-Π
7	Χ	Β-ΛΠ
8	Π	Β-ΛΧ
9	Λ	Β-ΠΧ
10		Β-ΛΠΧ

ΕΠΙΤΡΕΠΤΕΣ ΚΑΤΑΣΤΑΣΕΙΣ

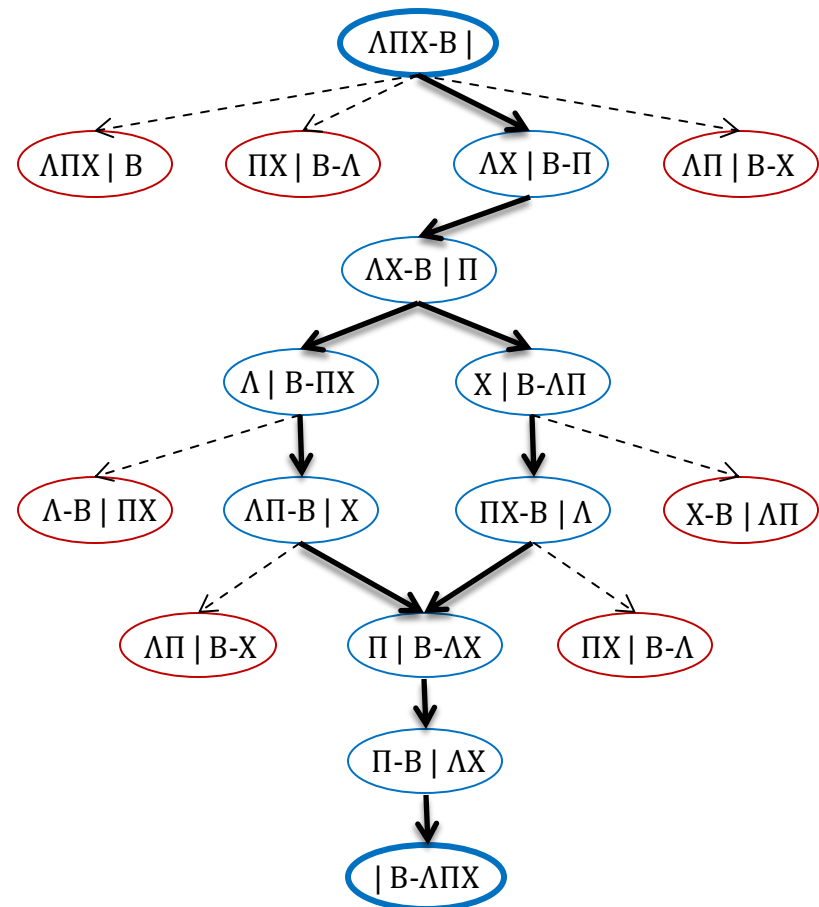


Το Πρόβλημα του Βαρκάρη : μετατροπή σε εξερεύνηση

■ **Πρόβλημα Βαρκάρη = Εύρεση διαδρομής σε γράφο**

1	ΛΠΧ-Β	
2	ΠΧ-Β	Λ
3	ΛΧ-Β	Π
4	ΛΠ-Β	Χ
5	Π-Β	ΛΧ
6	ΛΧ	Β-Π
7	Χ	Β-ΛΠ
8	Π	Β-ΛΧ
9	Λ	Β-ΠΧ
10		Β-ΛΠΧ

ΕΠΙΤΡΕΠΤΕΣ ΚΑΤΑΣΤΑΣΕΙΣ

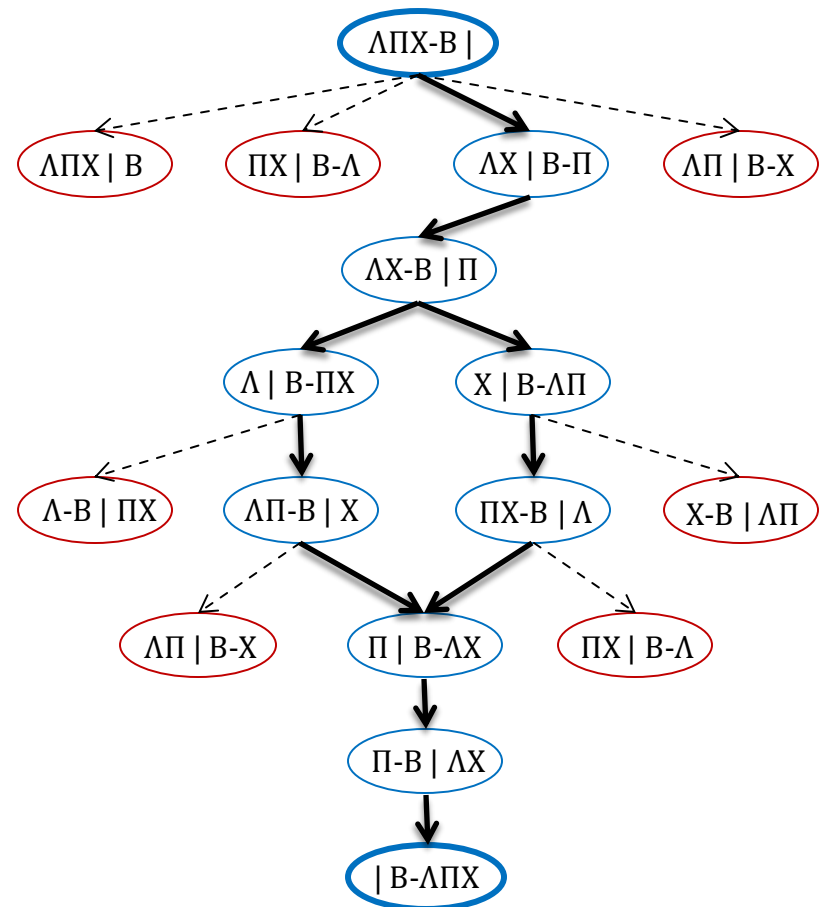


Το Πρόβλημα του Βαρκάρη : μετατροπή σε εξερεύνηση

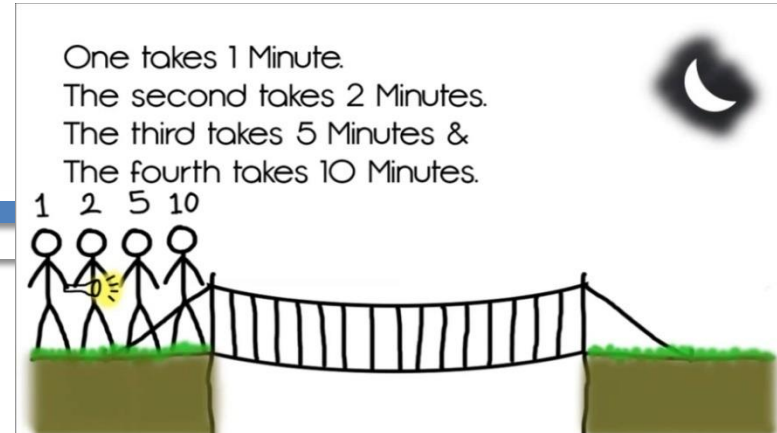
■ Μπορούμε να εφαρμόσουμε και σε άλλα προβλήματα;

1	ΛΠΧ-Β	
2	ΠΧ-Β	Λ
3	ΛΧ-Β	Π
4	ΛΠ-Β	Χ
5	Π-Β	ΛΧ
6	ΛΧ	Β-Π
7	Χ	Β-ΛΠ
8	Π	Β-ΛΧ
9	Λ	Β-ΠΧ
10		Β-ΛΠΧ

ΕΠΙΤΡΕΠΤΕΣ ΚΑΤΑΣΤΑΣΕΙΣ



Το Πρόβλημα της Γέφυρας



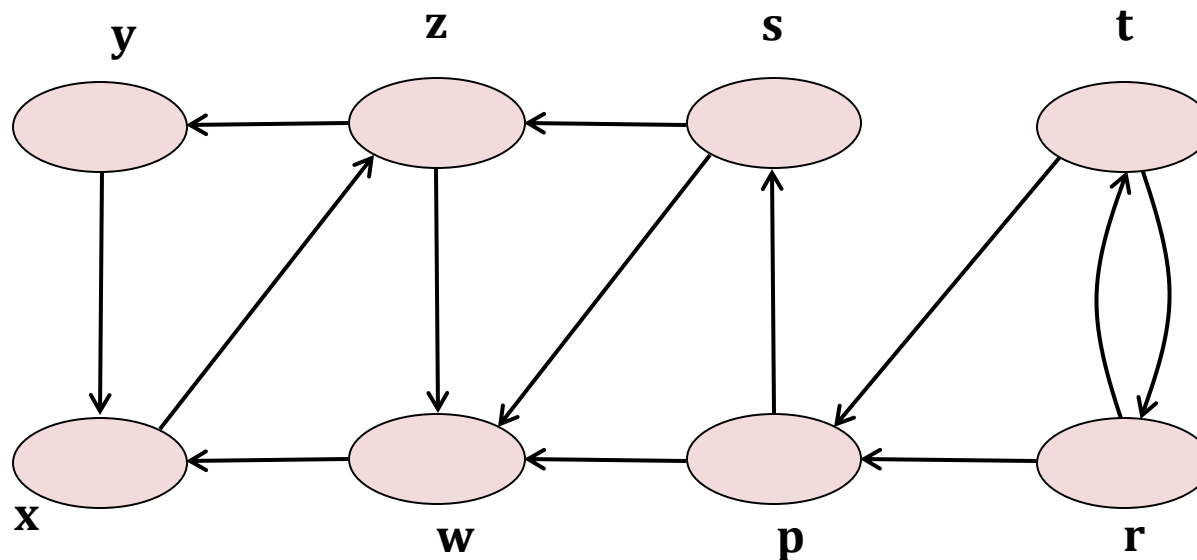
- ☐ 4 φίλοι (οι **A, B, Γ**, και **Δ**)
- ☐ θέλουν να διασχίσουν μια γέφυρα
- ☐ Η γέφυρα αντέχει μόνο 2 άτομα κάθε φορά
- ☐ Είναι νύχτα, και έχουν μόνο έναν φακό
- ☐ Ο καθένας χρειάζεται διαφορετικό χρόνο να περάσει τη γέφυρα: **1, 2, 5**, και **10** λεπτά είναι ο χρόνος για τον **A, B, Γ**, και **Δ** αντίστοιχα.
- ☐ Πόσο γρήγορα μπορούν να περάσουν τη γέφυρα;
 - ☐ Ιδέα: φτιάξτε τον **γράφο καταστάσεων**!

Επιπλέον εφαρμογές DFS - BFS

- 
- DFS: Ισχυρά συνεκτικές συνιστώσες (SCCs)
 - DFS: Σημεία άρθρωσης (articulation points) και γέφυρες (bridges)
 - BFS: Χωροθέτηση υπηρεσιών
 - DFS & BFS: Λαβύρινθοι

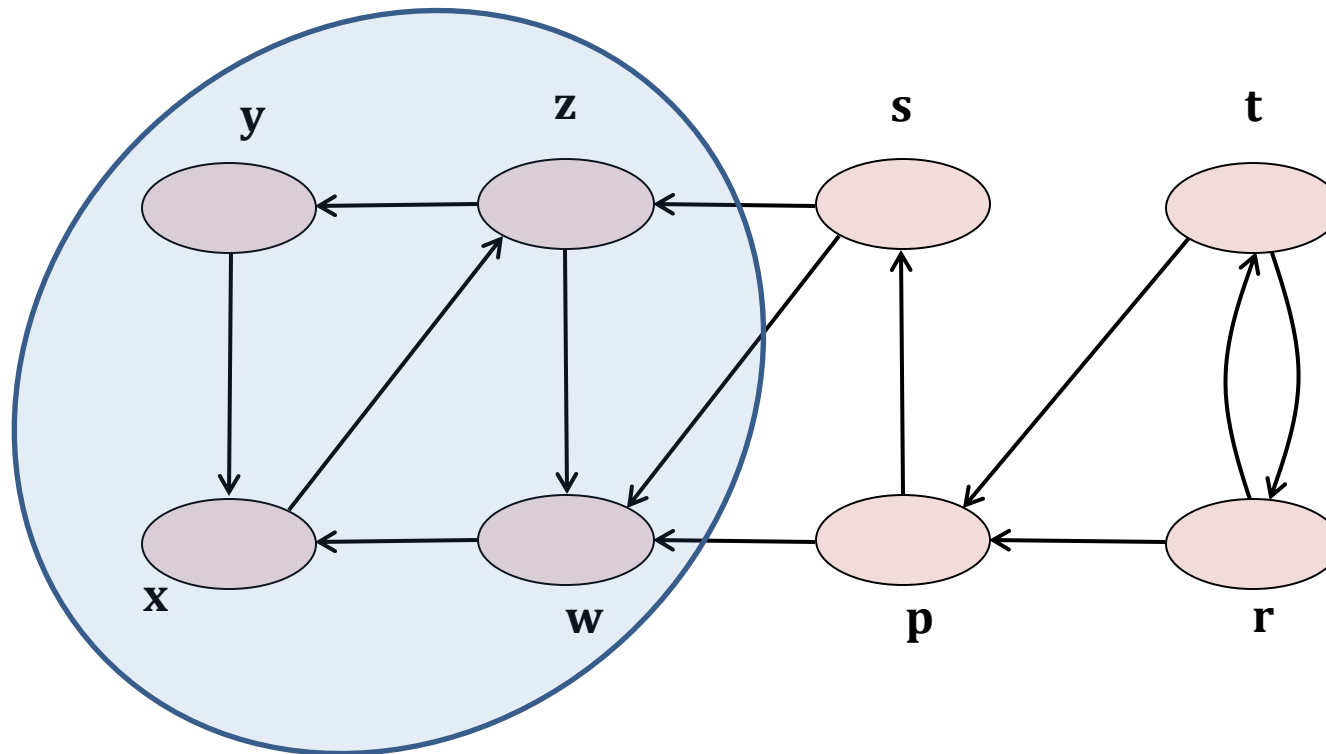
Ισχυρά συνεκτικές συνιστώσες (SCCs)

Ισχυρή συνεκτικότητα: υπάρχει διαδρομή από κάθε κόμβο σε κάθε άλλο κόμβο



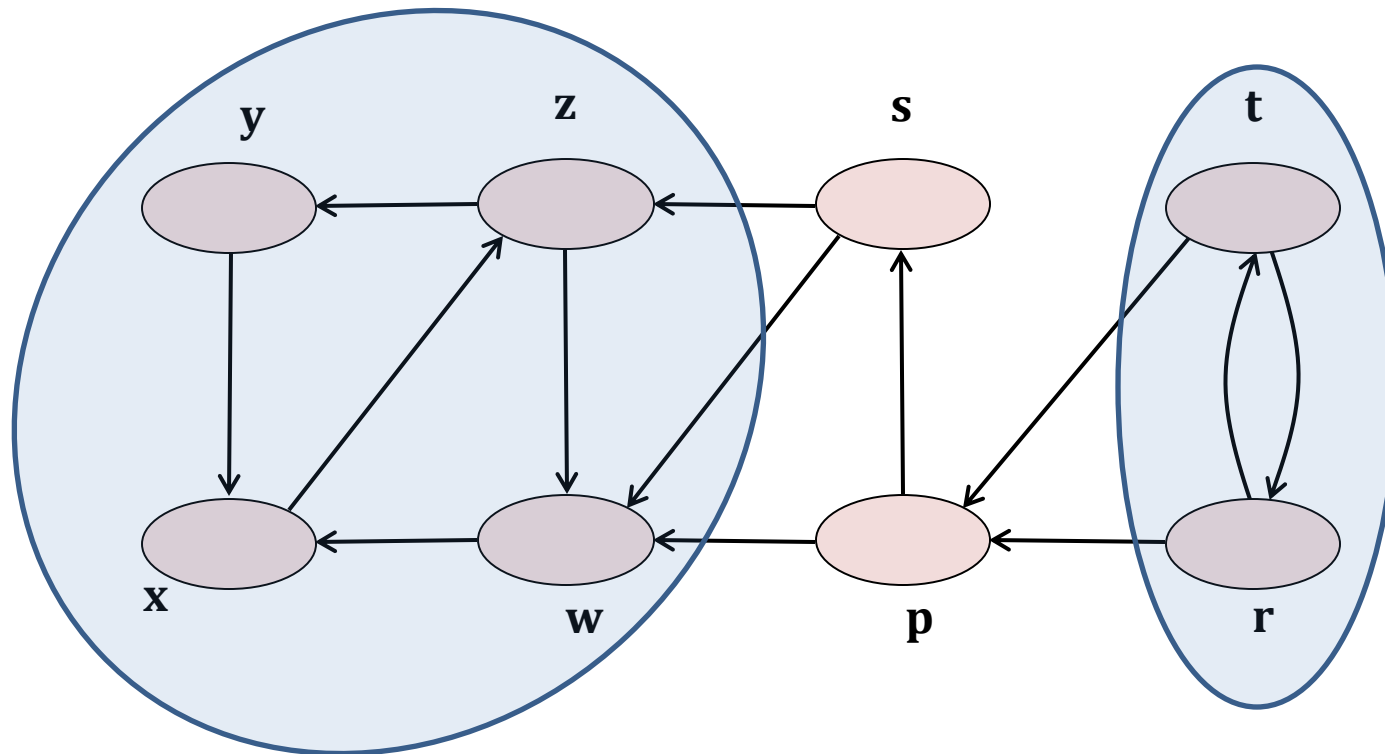
Πώς ελέγχουμε αν ο γράφος μας είναι ισχυρά συνεκτικός;

Ισχυρά συνεκτικές συνιστώσες (SCCs)



Μία συνεκτική συνιστώσα

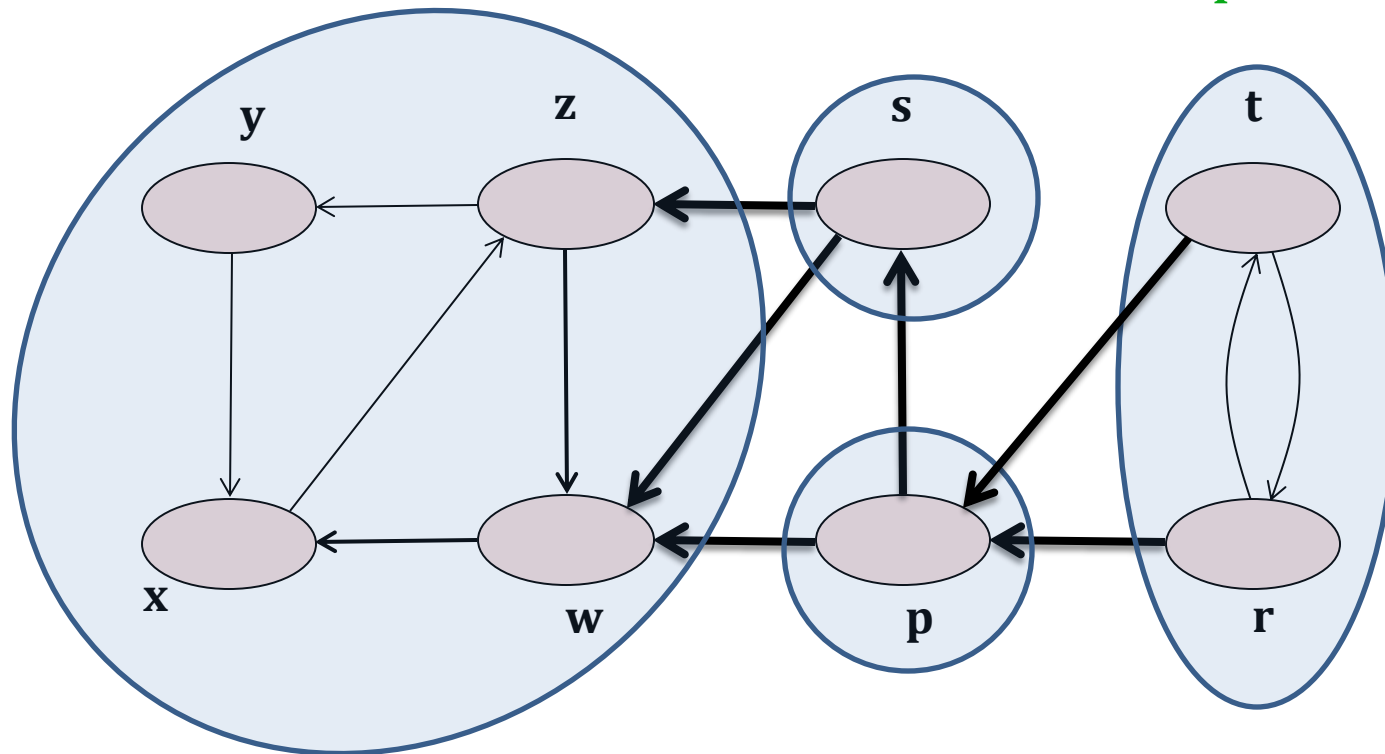
Ισχυρά συνεκτικές συνιστώσες (SCCs)



... κι άλλη μία!

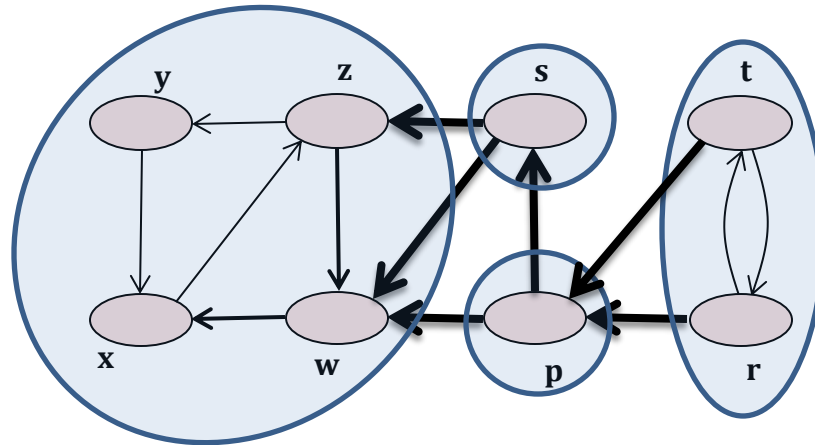
Ισχυρά συνεκτικές συνιστώσες (SCCs)

DAG of connected components



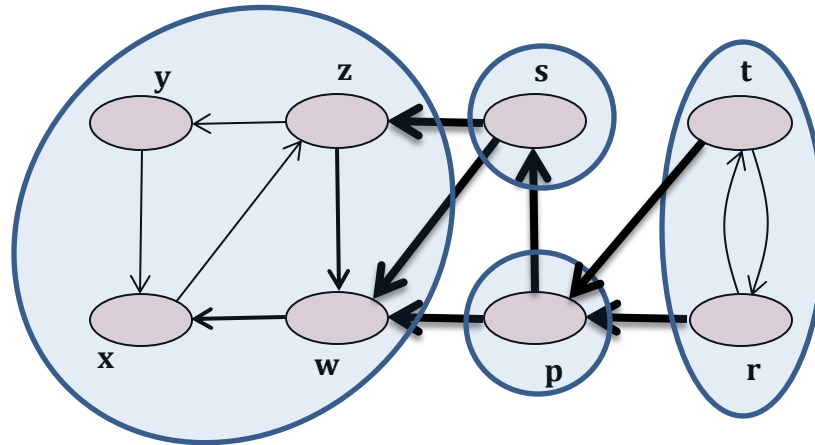
Πώς θα τις βρούμε όλες;

Ισχυρά συνεκτικές συνιστώσες (SCCs)



- Από πού πρέπει να ξεκινήσει η εξερεύνηση;
 - από συνιστώσα-προορισμό (sink)
- Πώς μπορούμε να βρούμε sink;
 - μπορούμε να βρούμε συνιστώσα-πηγή (source)... με DFS!
 - δουλεύουμε με αντεστραμμένο γράφο!!

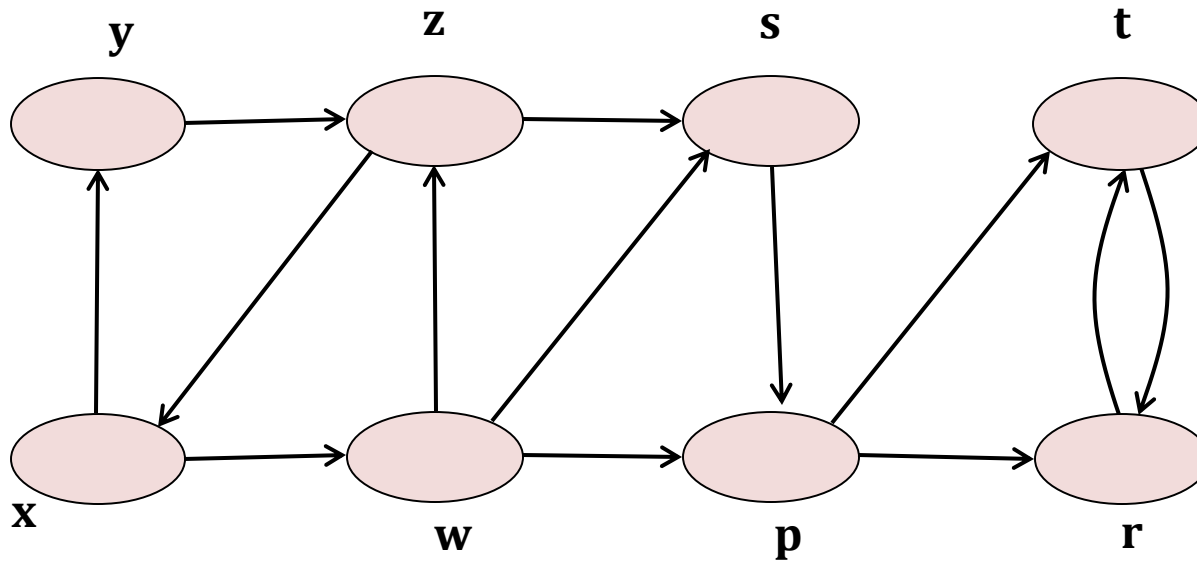
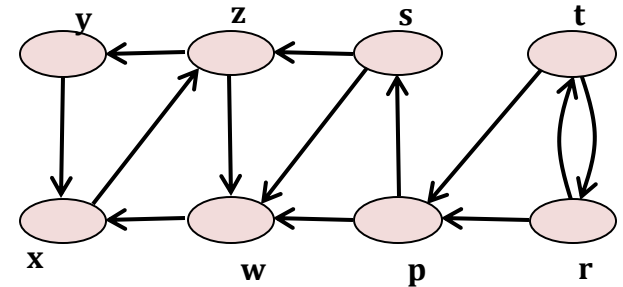
Ισχυρά συνεκτικές συνιστώσες (SCCs)



- Εκτέλεση DFS στον αντεστραμμένο γράφο G^R
- Ταξινόμηση κόμβων κατά φθίνουσα σειρά χρόνου αναχώρησης $t^R(v)$
- Εκτέλεση DFS στον G , επιλέγοντας κόμβους εκκίνησης με βάση την παραπάνω ταξινόμηση

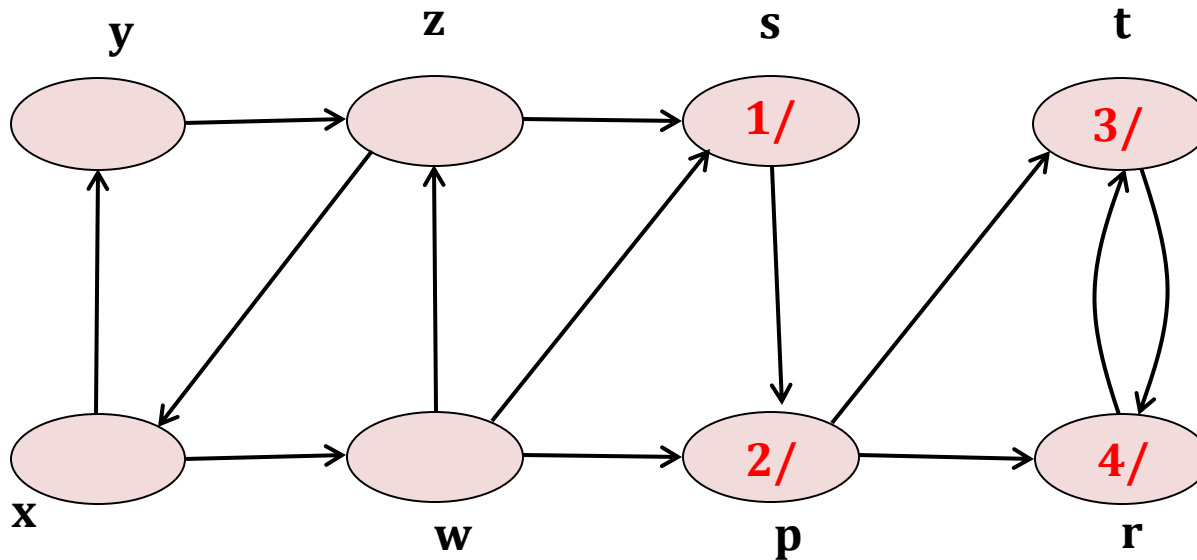
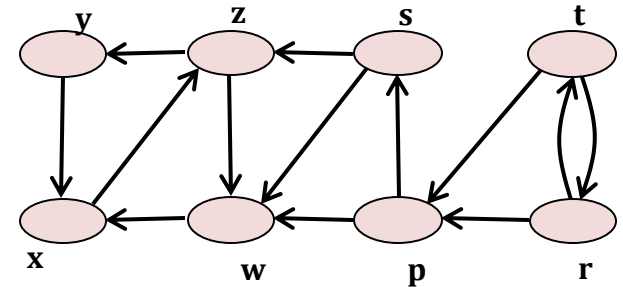
Ισχυρά συνεκτικές συνιστώσες (SCCs)

Παράδειγμα εκτέλεσης



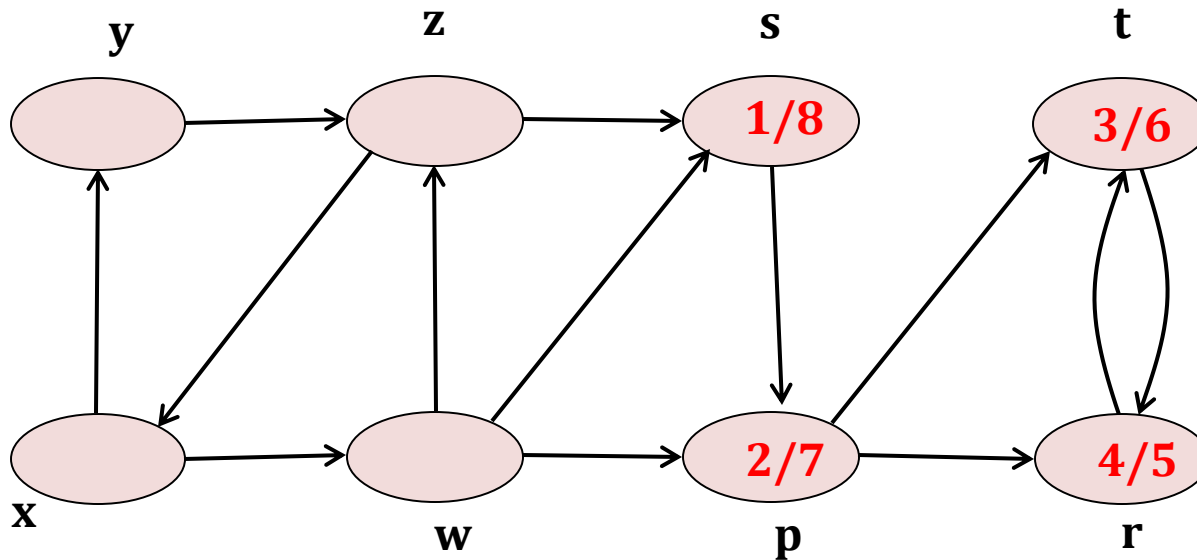
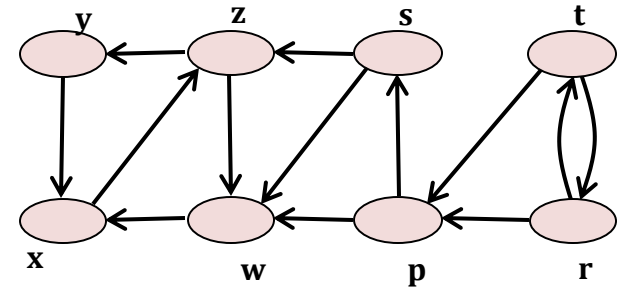
Ισχυρά συνεκτικές συνιστώσες (SCCs)

Παράδειγμα εκτέλεσης



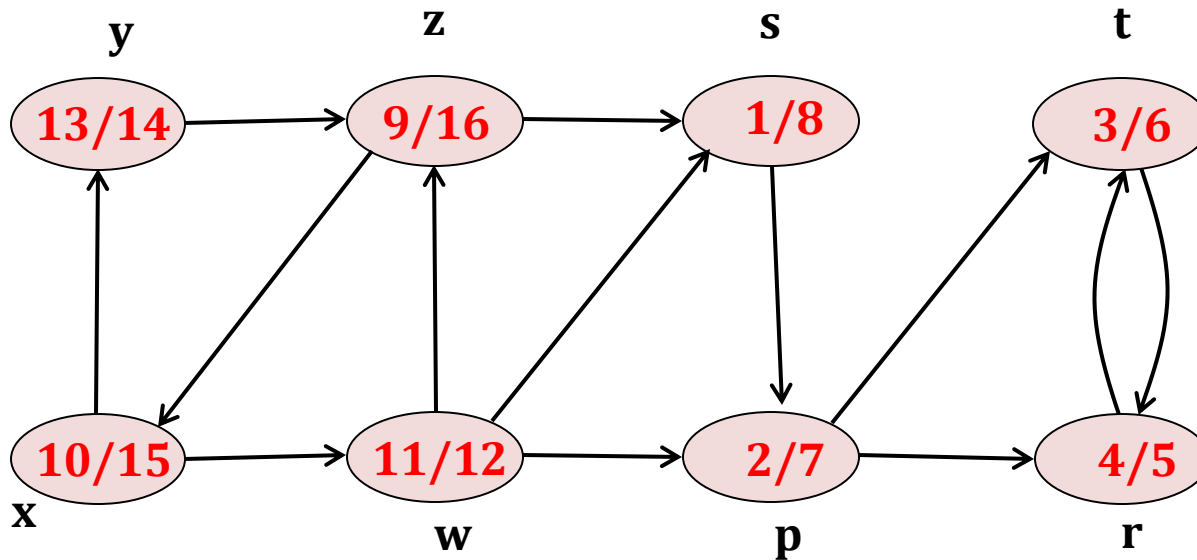
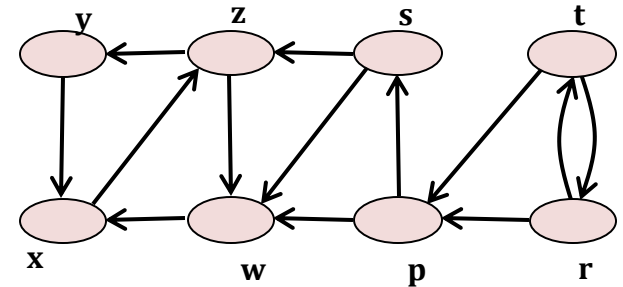
Ισχυρά συνεκτικές συνιστώσες (SCCs)

Παράδειγμα εκτέλεσης



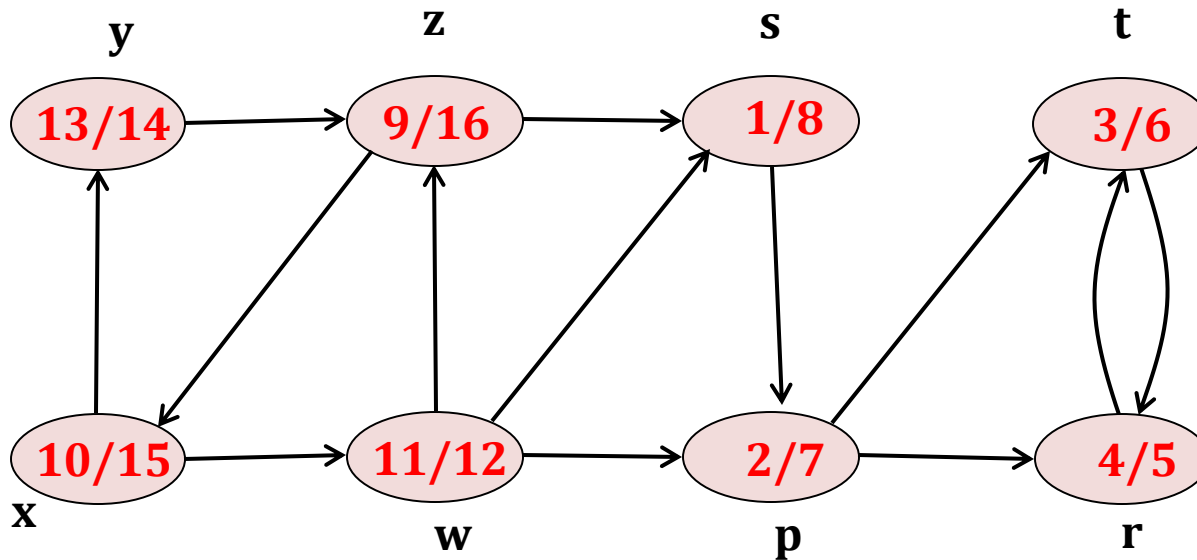
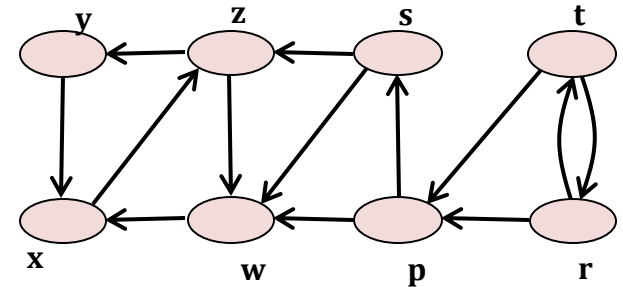
Ισχυρά συνεκτικές συνιστώσες (SCCs)

Παράδειγμα εκτέλεσης



Ισχυρά συνεκτικές συνιστώσες (SCCs)

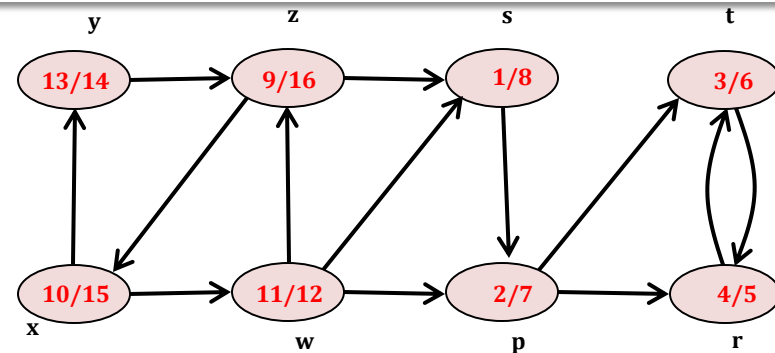
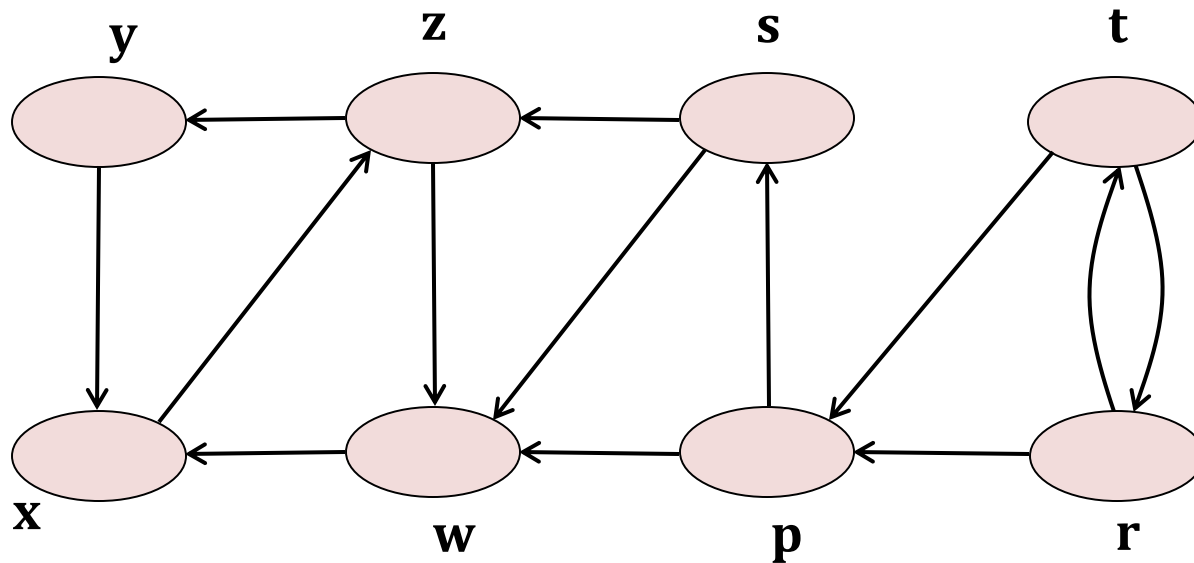
Παράδειγμα εκτέλεσης



Διάταξη κατά φθίνον t^R : z x y w s p t r

Ισχυρά συνεκτικές συνιστώσες (SCCs)

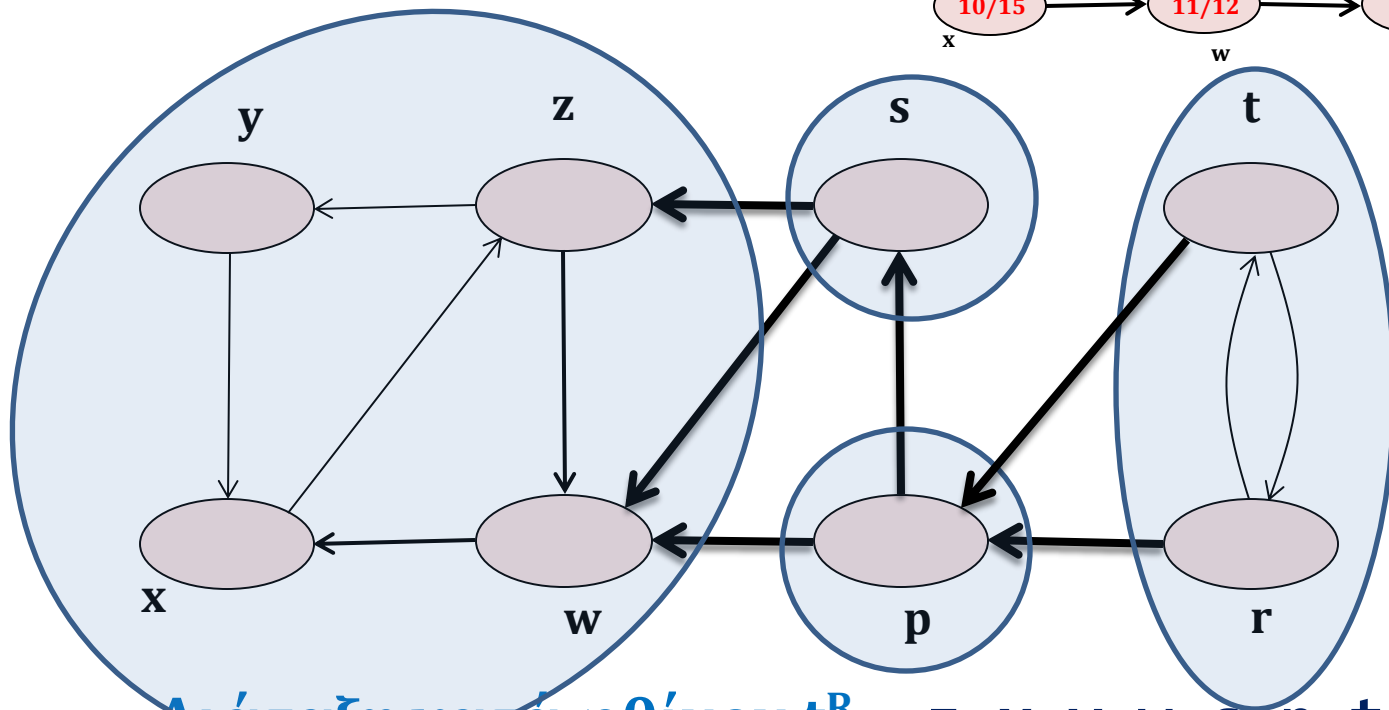
Παράδειγμα εκτέλεσης



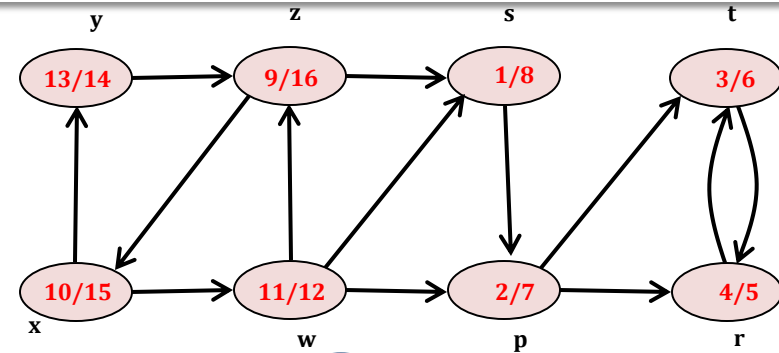
Διάταξη κατά φθίνον t^R : z x y w s p t r

Ισχυρά συνεκτικές συνιστώσες (SCCs)

Παράδειγμα εκτέλεσης



Διάταξη κατά φθίνον t^R : z x y w s p t r

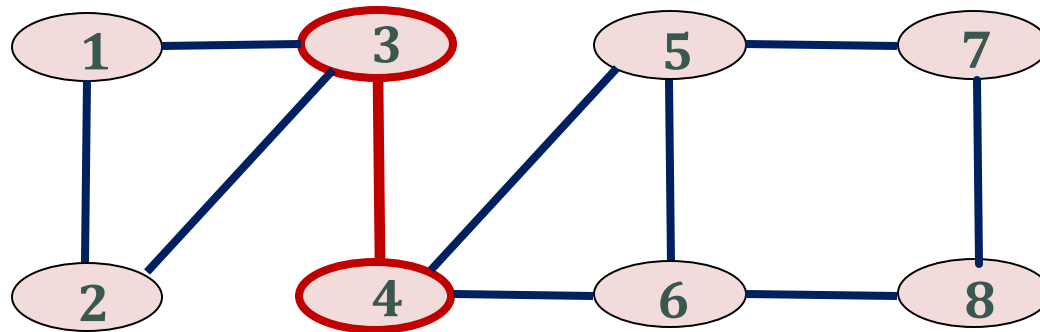


Εφαρμογές DFS

Ε

Σημεία άρθρωσης (articulation points): κορυφές που η αφαίρεσή τους αποσυνδέει τον γράφο

Γέφυρες (bridges): ακμές που η αφαίρεσή τους αποσυνδέει τον γράφο



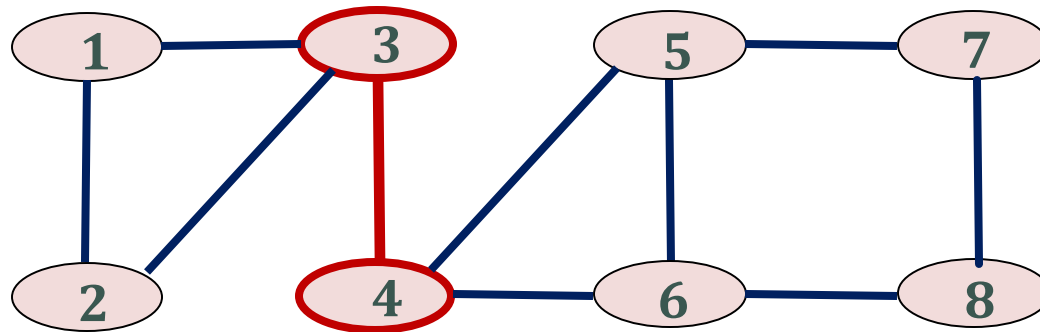
Ορίζεται σε **μη κατευθυνόμενους** γράφους

Εφαρμογές DFS

Ε

Σημεία άρθρωσης (articulation points): κορυφές που η αφαίρεσή τους αποσυνδέει τον γράφο

Γέφυρες (bridges): ακμές που η αφαίρεσή τους αποσυνδέει τον γράφο



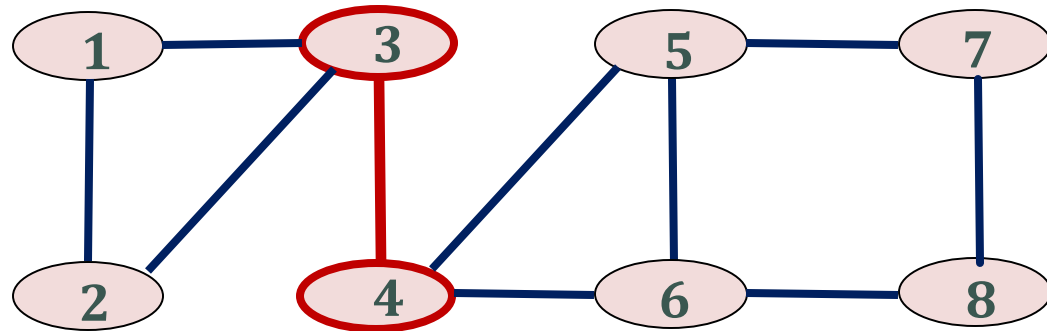
1 εκτέλεση DFS αρκεί !!

Εφαρμογές DFS

Ε Κριτήριο σημείων άρθρωσης με βάση δένδρο DFS

Σημείο άρθρωσης είναι:

- Η ρίζα αν έχει τουλάχιστον 2 παιδιά

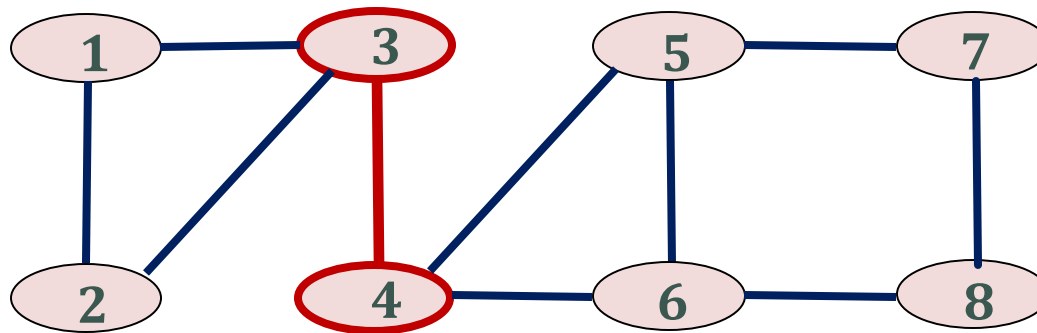


Εφαρμογές DFS

Ε Κριτήριο σημείων άρθρωσης με βάση δένδρο DFS

Σημείο άρθρωσης είναι:

- Η ρίζα αν έχει τουλάχιστον 2 παιδιά
- Κάθε εσωτερικός κόμβος v που έχει τουλάχιστον ένα παιδί με υποδένδρο που δεν έχει back edges σε κόμβους που προηγούνται του v

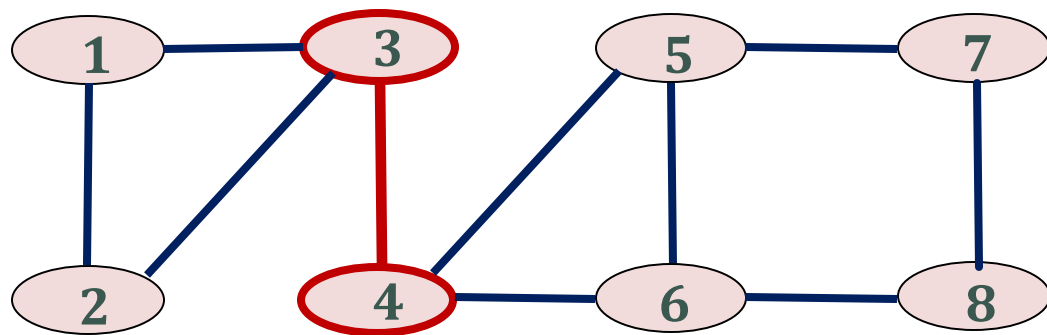


Εφαρμογές DFS

Ε Κριτήριο γέφυρας με βάση δένδρο DFS

Γέφυρα είναι μια ακμή (u,v) αν είναι δενδρική και:

- Το ένα άκρο της είναι «φύλλο» ή
- Τα δύο άκρα της u,v είναι σημεία άρθρωσης

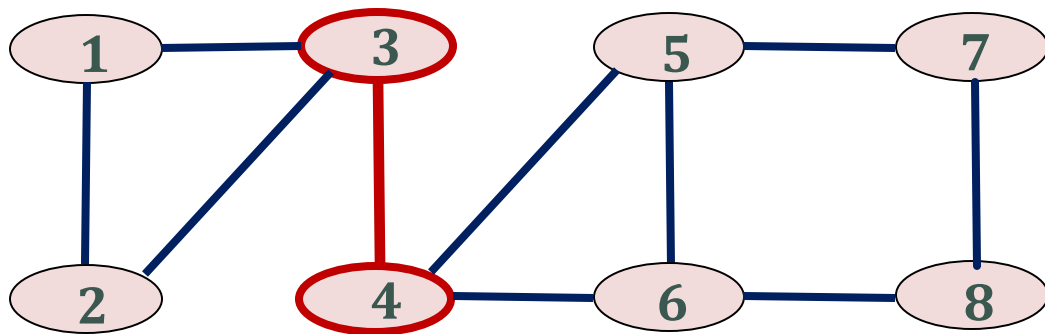


Εφαρμογές DFS

Ε Κριτήριο γέφυρας με βάση δένδρο DFS

Γέφυρα είναι μια ακμή (u,v) αν είναι δενδρική και:

- Το ένα άκρο της είναι «φύλλο» ή
- Τα δύο άκρα της u,v είναι σημεία άρθρωσης **και** στο άκρο που βρέθηκε δεύτερο (έστω v) κανένας κόμβος του υποδένδρου του δεν έχει back edge προς το άκρο που βρέθηκε πρώτο (u)

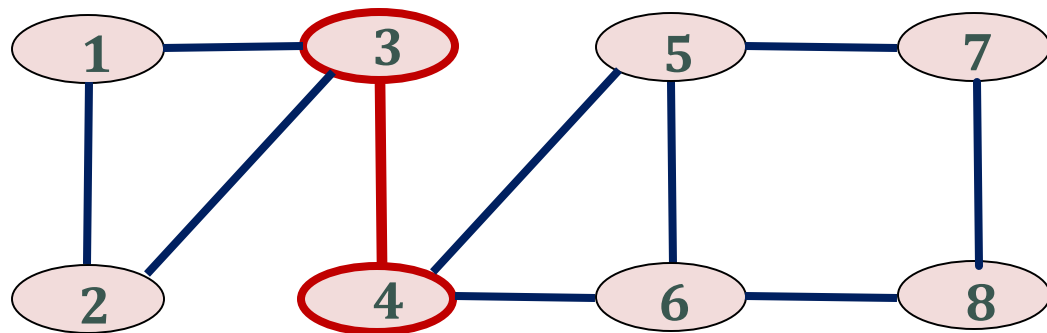


Εφαρμογές DFS

Ε Ισοδύναμο κριτήριο γέφυρας με βάση δένδρο DFS

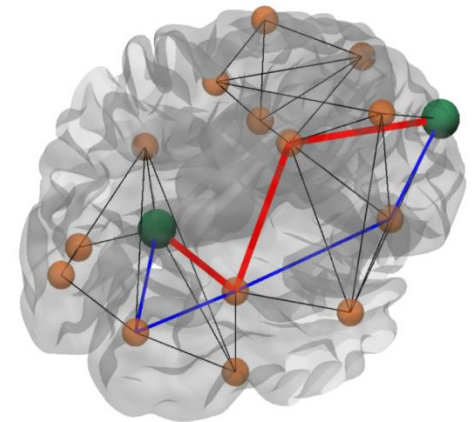
Γέφυρα είναι μια ακμή (u,v) αν είναι δενδρική και:

- Στο άκρο που βρέθηκε δεύτερο (έστω v) κανένας κόμβος του υποδένδρου του δεν έχει back edge προς το άκρο που βρέθηκε πρώτο (u) ούτε σε πρόγονο του u .



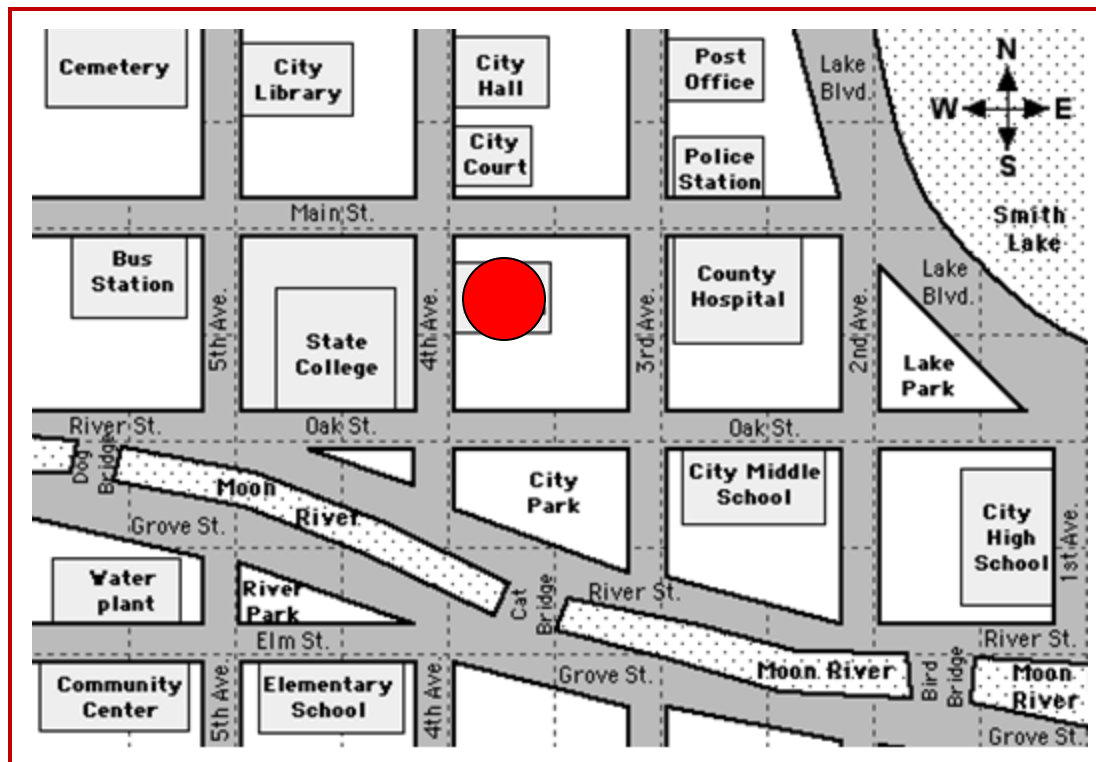
Εφαρμογές BFS

Ε Χωροθέτηση Υπηρεσιών και Λαβύρινθοι



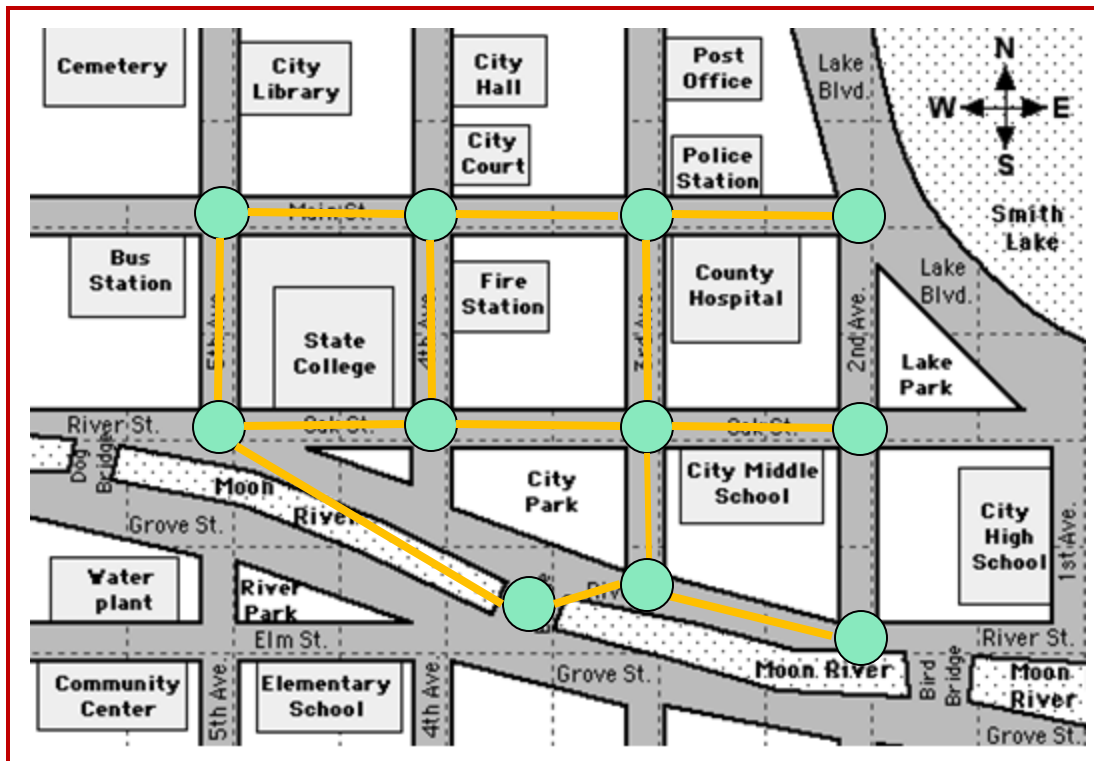
Εφαρμογές BFS

Ε Χωροθέτηση Πυροσβεστικού Τμήματος



Εφαρμογές BFS

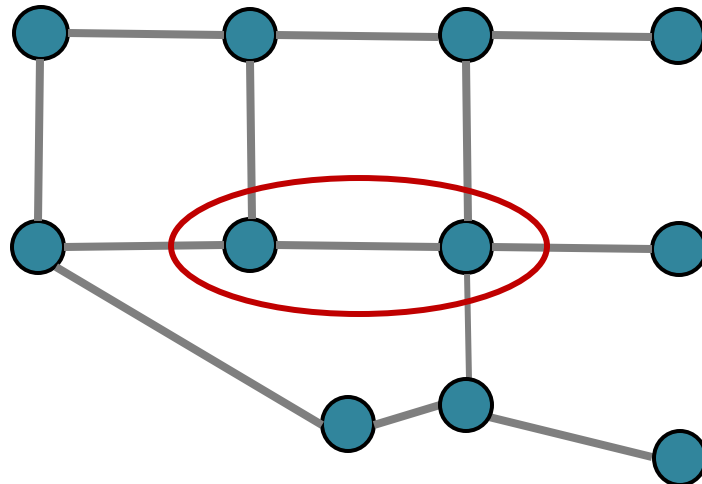
Ε Χωροθέτηση Πυροσβεστικού Τμήματος



Εφαρμογές BFS

Ε Χωροθέτηση Πυροσβεστικού Τμήματος

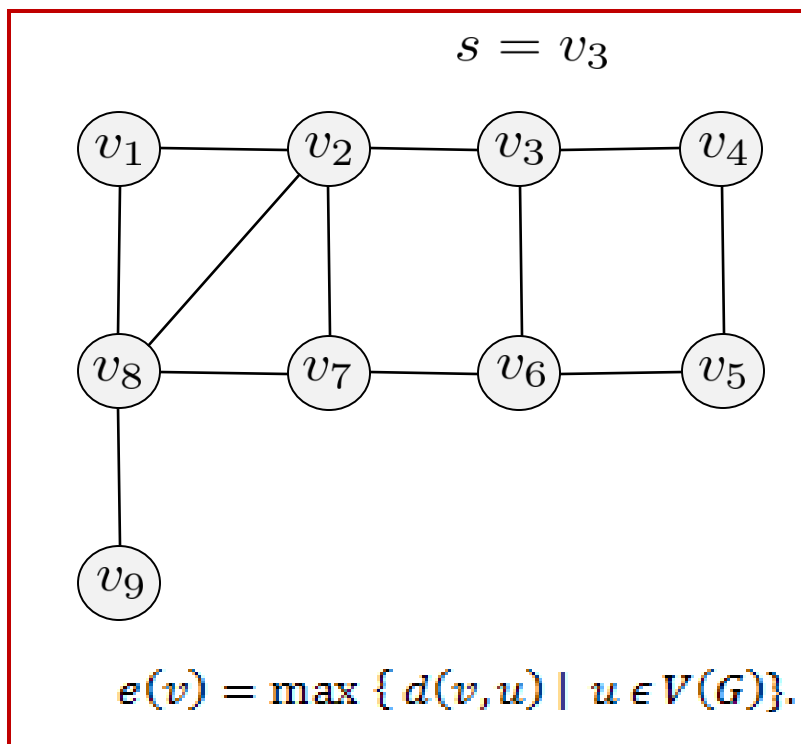
Ζητείται ελάχιστη εκκεντρότητα (eccentricity)



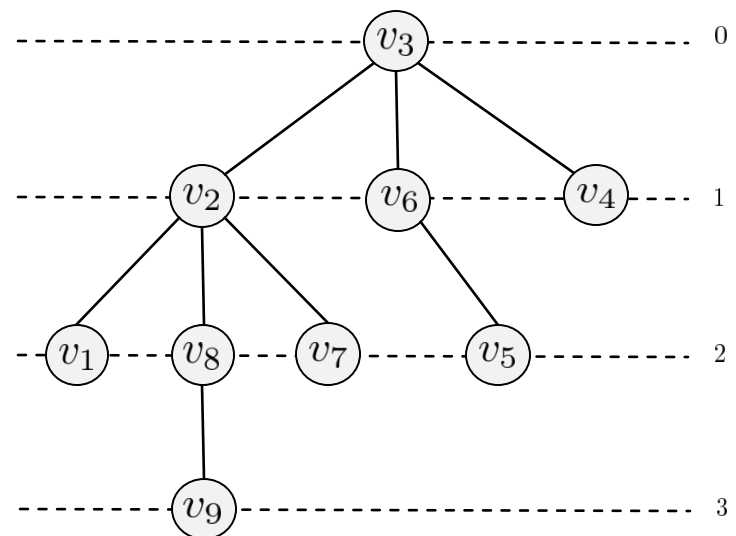
$$e(v) = \max \{ d(v, u) \mid u \in V(G) \}.$$

Εφαρμογές BFS

Ε Χωροθέτηση Πυροσβεστικού Τμήματος



BFS-δένδρο



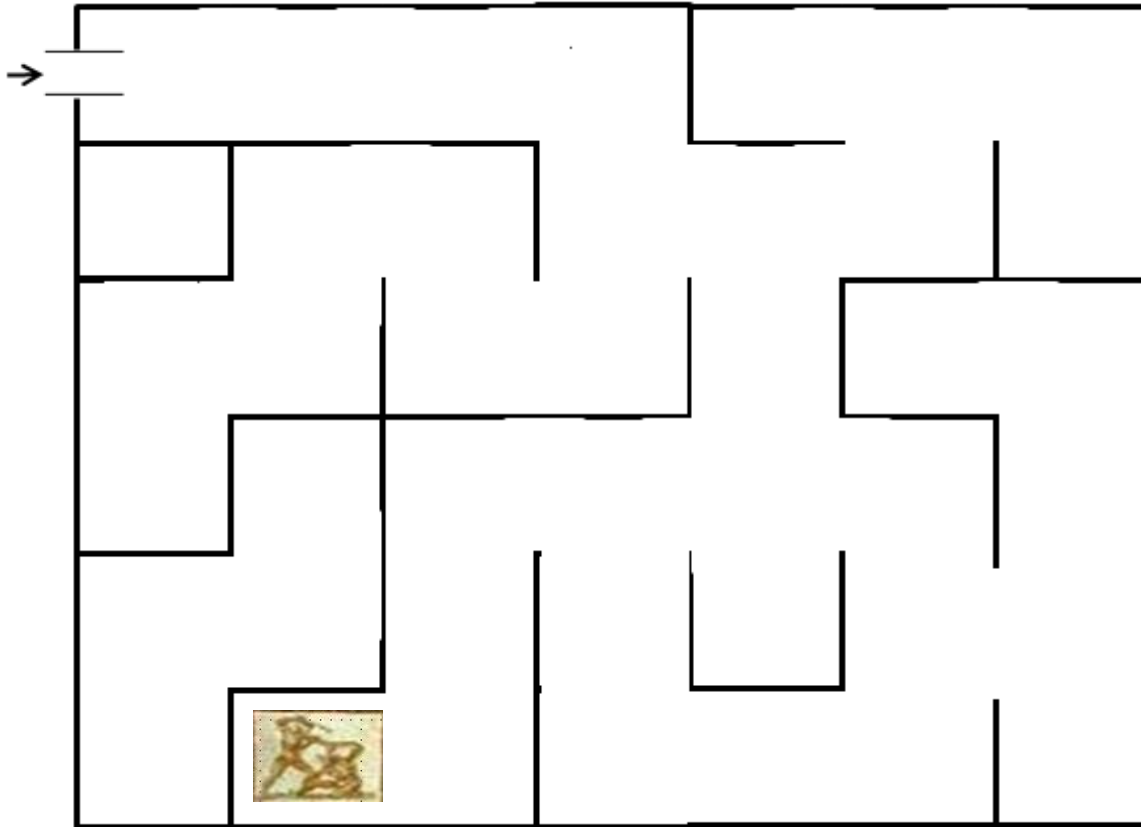
Εφαρμογές BFS & DFS

Ε Εετυλίγοντας τον Μίτο της Αριάδνης!



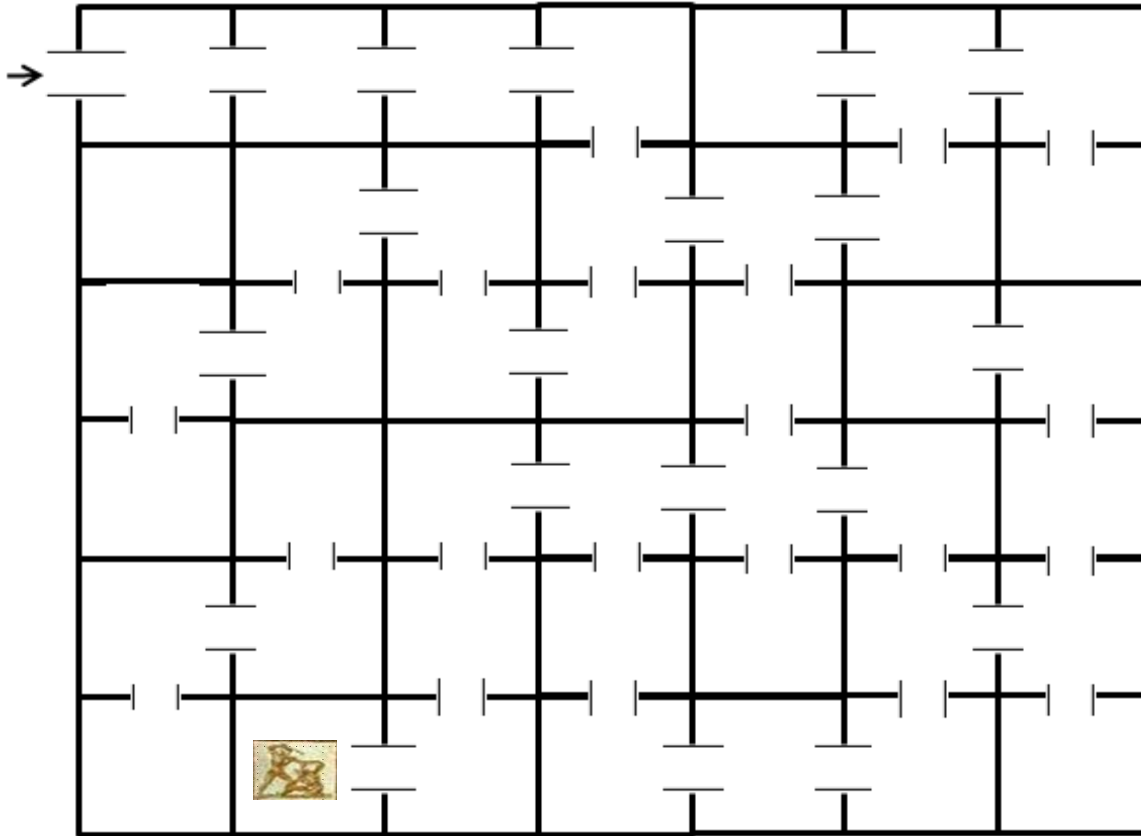
Εφαρμογές BFS & DFS

Ε Εετυλίγοντας τον Μίτο της Αριάδνης!



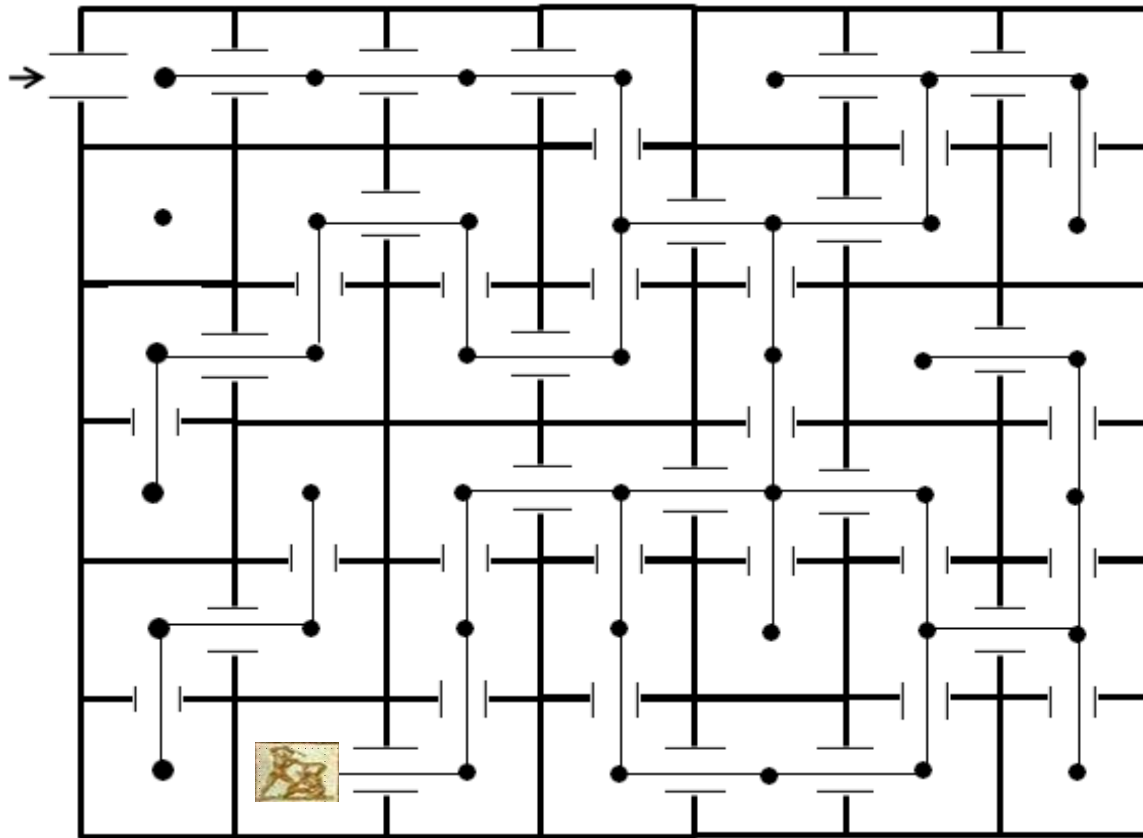
Εφαρμογές BFS & DFS

Εξευλίγοντας τον Μίτο της Αριάδνης!



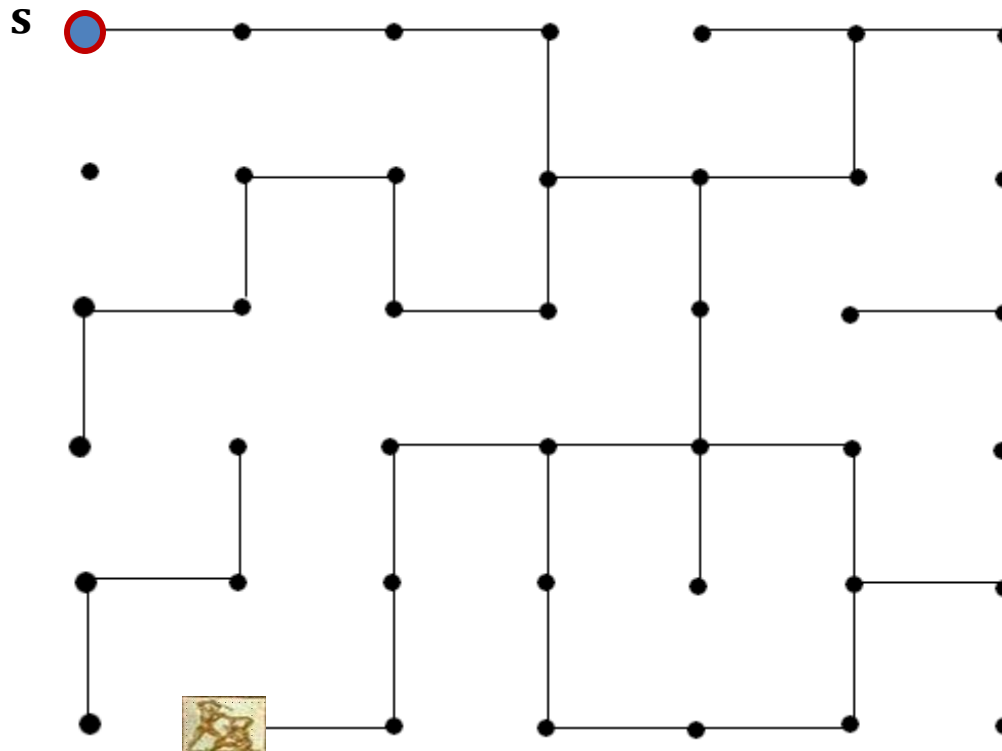
Εφαρμογές BFS & DFS

Ε Εετυλίγοντας τον Μίτο της Αριάδνης!

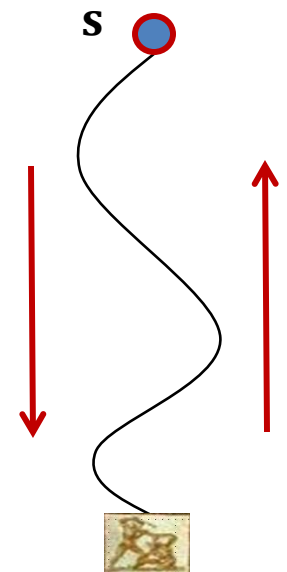


Εφαρμογές BFS & DFS

Ε Ξετυλίζοντας τον Μίτο της Αριάδνης!



DFS-εξερεύνηση



πότε προτιμούμε *DFS*;