Δομές Δεδομένων

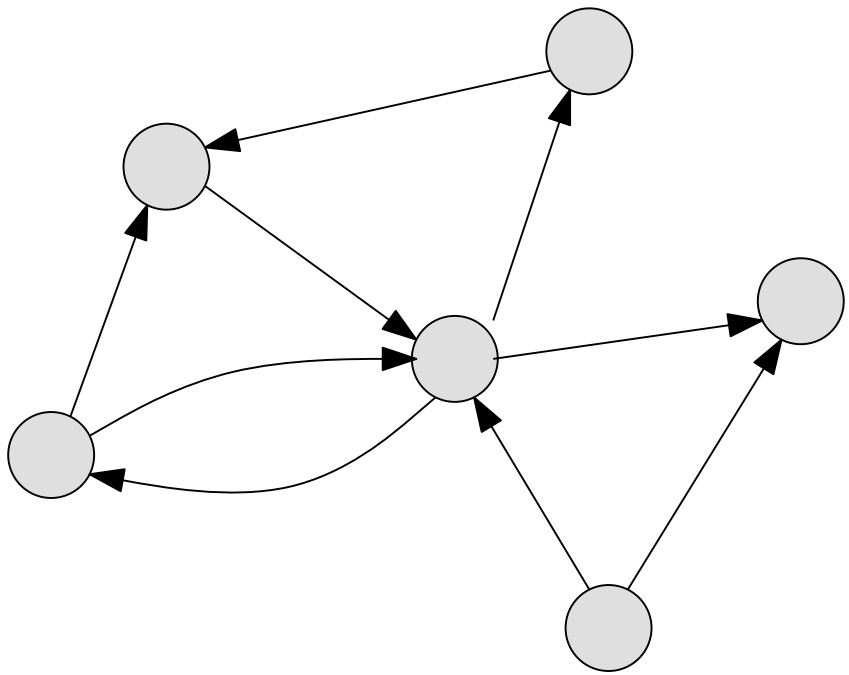
Γραφήματα

Δημήτρης Μιχαήλ

Τμήμα Πληροφορικής και Τηλεματικής
Χαροκόπειο Πανεπιστήμιο

Κατευθυνόμενο Γράφημα

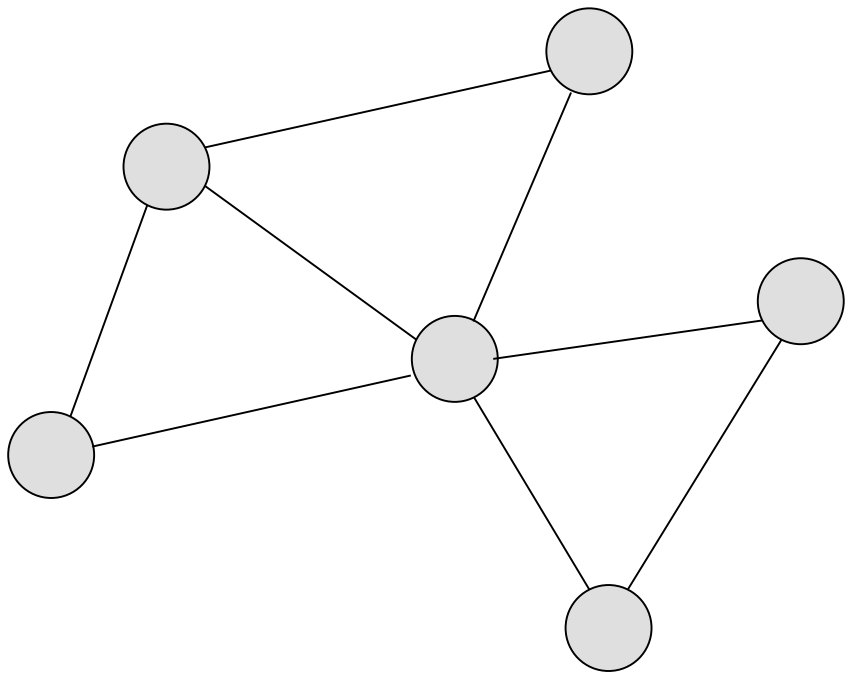
Ένα κατευθυνόμενο γράφημα G είναι ένα ζευγάρι (V, E) όπου V είναι ένα σύνολο κόμβων και E είναι ένα σύνολο με διατεταγμένα ζευγάρια κόμβων.



Γραφήματα

Μη-κατευθυνόμενο Γράφημα

Ένα μη-κατευθυνόμενο γράφημα G είναι ένα ζευγάρι (V, E) όπου V είναι ένα σύνολο κόμβων και E είναι ένα σύνολο με ζευγάρια κόμβων.



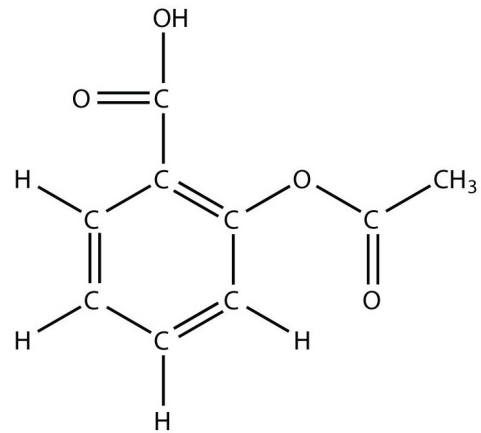
Μοντελοποίηση Προβλημάτων

Πάρα πολλά προβλήματα μοντελοποιούνται με γραφήματα.

Δίκτυο δρόμων

Κοινωνικά Δίκτυα

Μόρια



Βασικές Λειτουργίες

- **Προσπέλαση συσχετιζόμενων πληροφοριών.**
  Δεδομένου ενός κόμβου ή ακμής, προσπέλαση πληροφορίας που σχετίζεται, π.χ βάρος ακμής, απόσταση από άλλο κόμβο
- **Πλοήγηση.**
  - Με δεδομένο έναν κόμβο, προσπέλαση των εξερχόμενων ακμών του.
  - Αυτή η λειτουργία βρίσκεται στην καρδιά των περισσότερων αλγορίθμων σε γραφήματα.
  - Μερικές φορές θέλουμε εύκολη προσπέλαση και των εισερχόμενων ακμών ενός κόμβου.
- **Ερωτήματα ακμής.**
  - Με δεδομένο 2 κόμβους (u, v) θέλουμε να ξέρουμε αν η συγκεκριμένη ακμή υπάρχει σε ένα γράφημα.
  - Μερικές φορές θέλουμε να μπορούμε να βρούμε εύκολα την αντίθετη ακμή (v, u) μίας κατευθυνόμενης ακμής (u, v).
- **Κατασκευή, μετατροπή και έξοδος.** Μετατροπή της αναπαράστασης σε πιο κατάλληλη για το συγκεκριμένο αλγοριθμικό πρόβλημα που θέλουμε να λύσουμε.
- **Ενημέρωση.** Προσθήκη και διαγραφή κόμβων ή ακμών, αλλαγή πληροφορίας κόμβου ή ακμής.

Αναπαράσταση Γραφημάτων

Υπάρχουν διάφορες μέθοδοι, ανάλογα με τις ανάγκες μας σε χρόνο, χώρο αλλά και κατά πόσο θέλουμε να υποστηρίξουμε δυναμικά ή μόνο στατικά γραφήματα.

- Μη-διατεταγμένη ακολουθία ακμών
- Πίνακες γειτνίασης (στατικά γραφήματα)
- Μητρώο γειτνίασης (adjacency matrix)
- Λίστες γειτνίασης (adjacency lists)

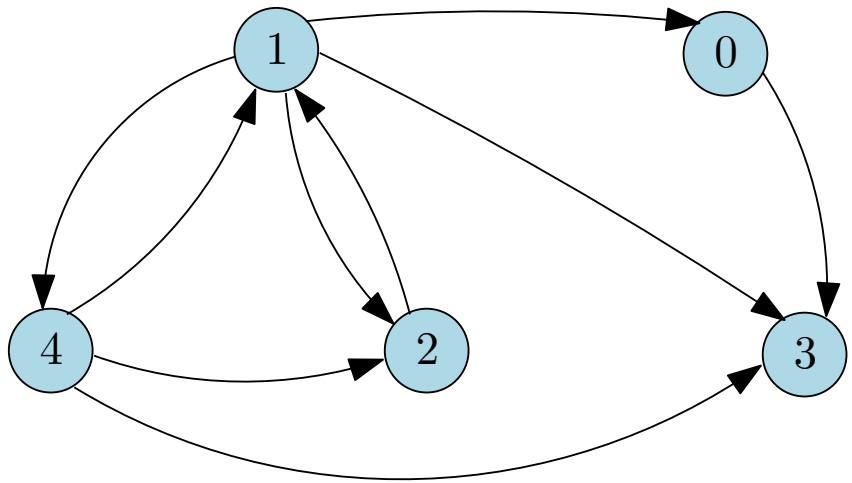
ή άλλες που μπορεί να είναι κατάλληλες ανάλογα με την εφαρμογή.

Μη διατεταγμένη ακολουθία ακμών

γράφημα $G(V, E)$ όπου $|V| = n$ και $|E| = m$

Η αναπαράσταση περιέχει:

- μία μη διατεταγμένη λίστα με τις ακμές



$$((1,0),(0,3),(1,3),(1,2),(4,3),(2,1),(4,2),(1,4),(4,1))$$

Μη διατεταγμένη ακολουθία ακμών

γράφημα $G(V, E)$ όπου $|V| = n$ και $|E| = m$

$$((1,0),(0,3),(1,3),(1,2),(4,3),(2,1),(4,2),(1,4),(4,1))$$

Χαρακτηριστικά

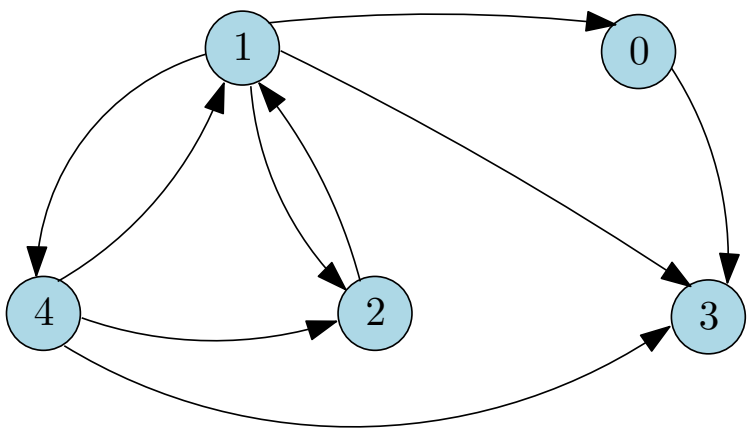
- συνήθης μορφή για είσοδο/έξοδο
- εύκολη προσθήκη ακμής ή κόμβου σε $\mathcal{O}(1)$
- όλες η υπόλοιπες (π.χ πλοήγηση) $\mathcal{O}(m)$
  - απαγορευτικά μεγάλος

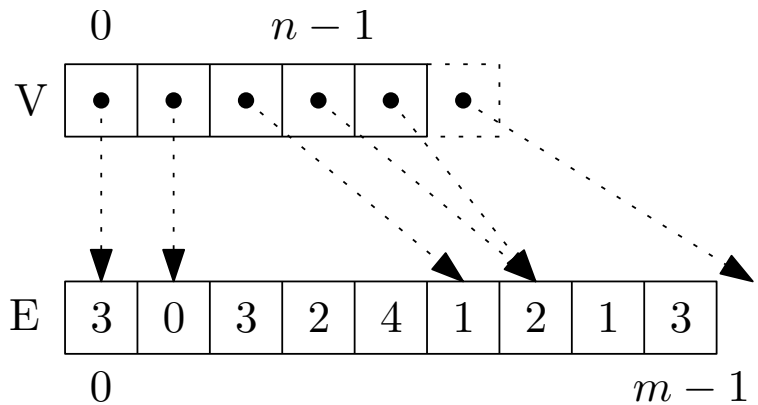
Πίνακας Γειτνίασης

γράφημα $G(V, E)$ όπου $|V| = n$ και $|E| = m$

Η αναπαράσταση περιέχει:

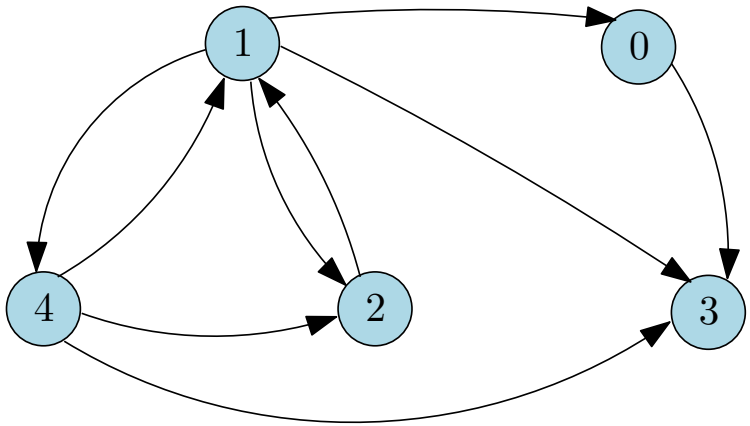
- πίνακας $E[0 \ldots m-1]$ με ομαδοποιημένες τις εξερχόμενες ακμές ανά κόμβο
- πίνακας $V[0 \ldots n-1]$ με θέσεις εκκίνησης υποπινάκων (ανά κόμβο)
  - Για κόμβο v, η θέση $V[v]$ περιέχει την θέση του E όπου βρίσκεται η πρώτη εξερχόμενη ακμή του v
  - Βολεύει να προσθέσουμε εικονική καταχώρηση $V[n] = m$, τότε οι . εξερχόμενες ακμές του v είναι οι $E[V[v]], \ldots, E[V[v+1]-1]$.

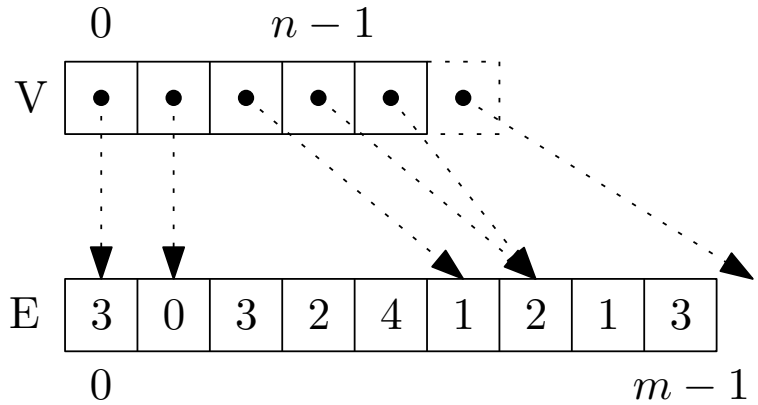




Πίνακας Γειτνίασης

γράφημα $G(V, E)$ όπου $|V| = n$ και $|E| = m$





Χαρακτηριστικά

- κατανάλωση χώρου $n + m + \Theta(1)$ λέξεις
- επιπλέον πληροφορίες μπορούν να αποθηκευθούν μέσα στον πίνακα ακμών
- στατική αναπαράσταση
- πλοήγηση (π.χ εκτύπωση εξερχόμενων ακμών) χρόνο ίσο με τον βαθμό του κόμβου $\mathcal{O}(d)$
- ερώτημα ακμής $\mathcal{O}(d)$ χρόνο

Πίνακας Γειτνίασης

Άσκηση

Σχεδιάστε έναν αλγόριθμο γραμμικού χρόνο $\mathcal{O}(n+m)$ ο οποίος πρέπει να μετατρέπει την αναπαράσταση ενός κατευθυνόμενου γραφήματος με μη-διατεταγμένη ακολουθία ακμών σε αναπαράσταση με πίνακα γειτνίασης. Πρέπει να χρησιμοποιήσετε μόνο $\mathcal{O}(1)$ βοηθητικό χώρο.

Υπόδειξη: Θεωρήστε πως το πρόβλημα είναι η ταξινόμηση των ακμών με βάση τον κόμβο προέλευσης τους. Χρησιμοποιήστε, προσαρμόζοντας κατάλληλα, τον αλγόριθμο ταξινόμηση ακεραίων που τρέχει σε γραμμικό χρόνο.

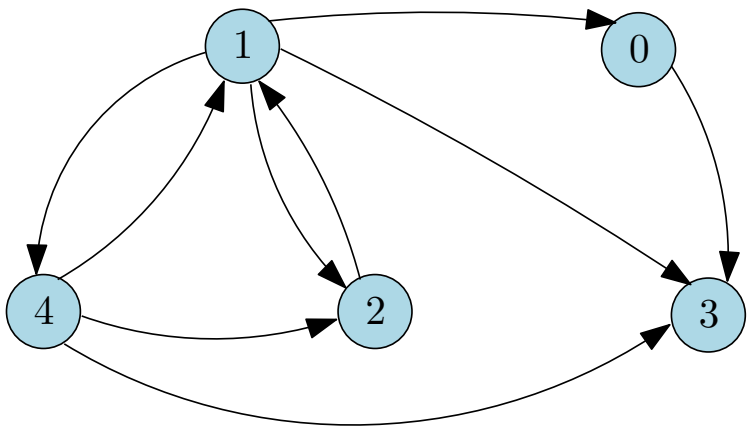
Θα χρειαστεί να διαβάσετε το κεφάλαιο ταξινόμηση.

Μητρώο γειτνίασης

γράφημα $G(V, E)$ όπου $|V| = n$ και $|E| = m$

Η αναπαράσταση περιέχει:

- ένα διδιάστατο πίνακα A με διαστάσεις $n \times n$
- η θέση του πίνακα i, j υποδηλώνει την ύπαρξη ή όχι της κατευθυνόμενης ακμής (i, j)
- εάν $e = (i, j) \in E$ τότε $A[i][j] = 1$ αλλιώς 0



$$A = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

Μητρώο γειτνίασης

γράφημα $G(V, E)$ όπου $|V| = n$ και $|E| = m$

Χαρακτηριστικά

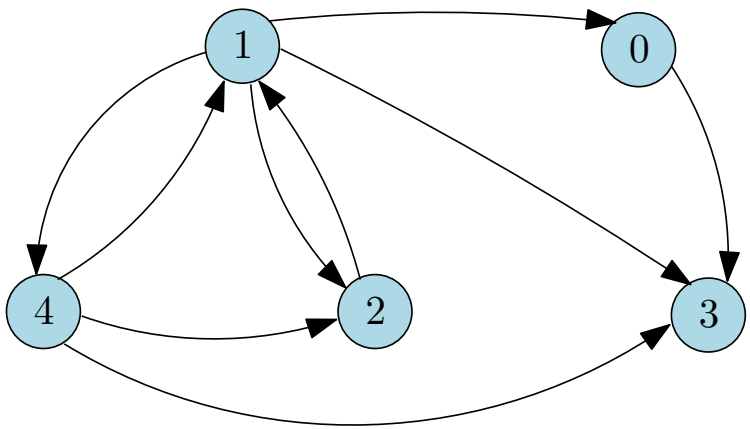
- Εισαγωγή/διαγραφή ακμών σε $\mathcal{O}(1)$ χρόνο
- Ερωτήματα ακμών σε $\mathcal{O}(1)$ χρόνο
- Πλοήγηση σε $\mathcal{O}(n)$ χρόνο - αποδοτικό μόνο για πολύ πυκνά γραφήματα
- Χώρος n^2 bits
- Εννοιολογική σύνδεση μεταξύ γραφημάτων και γραμμικής άλγεβρας, π.χ εαν $C = A^k$ τότε το C_{ij} μετρά τον συνολικό αριθμό των διαδρομών από τον κόμβο i στον κόμβο j οι οποίες έχουν k ακμές.

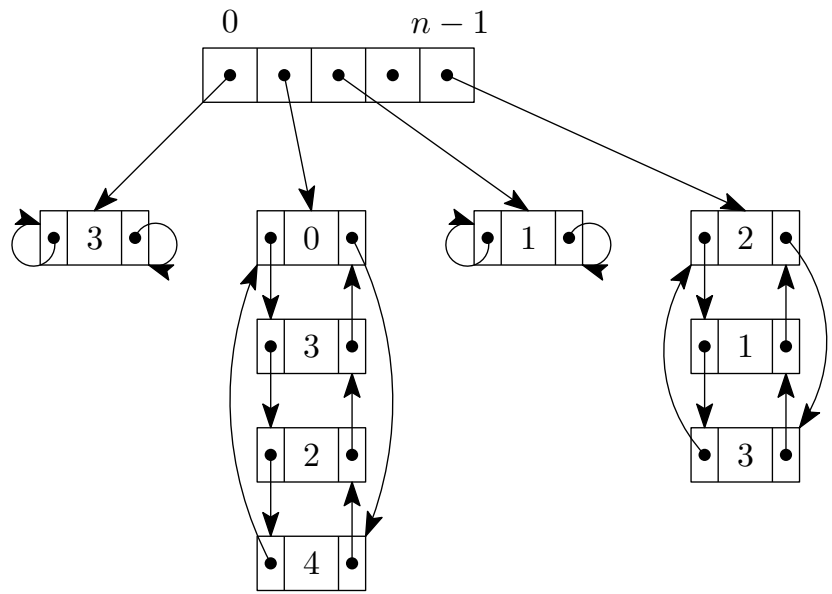
Λίστες γειτνίασης

γράφημα $G(V, E)$ όπου $|V| = n$ και $|E| = m$

Η αναπαράσταση περιέχει:

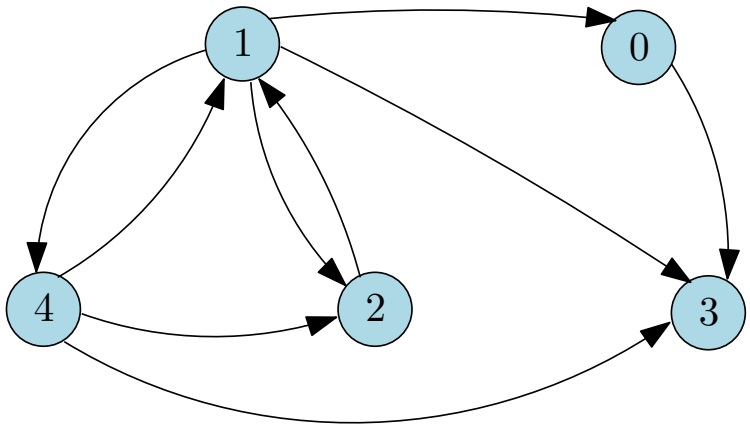
- ένα μονοδιάστατο πίνακα A με μία θέση για κάθε κόμβο
- κάθε θέση του πίνακα A είναι μία λίστα ακμών που εξέρχονται από τον αντίστοιχο κόμβο

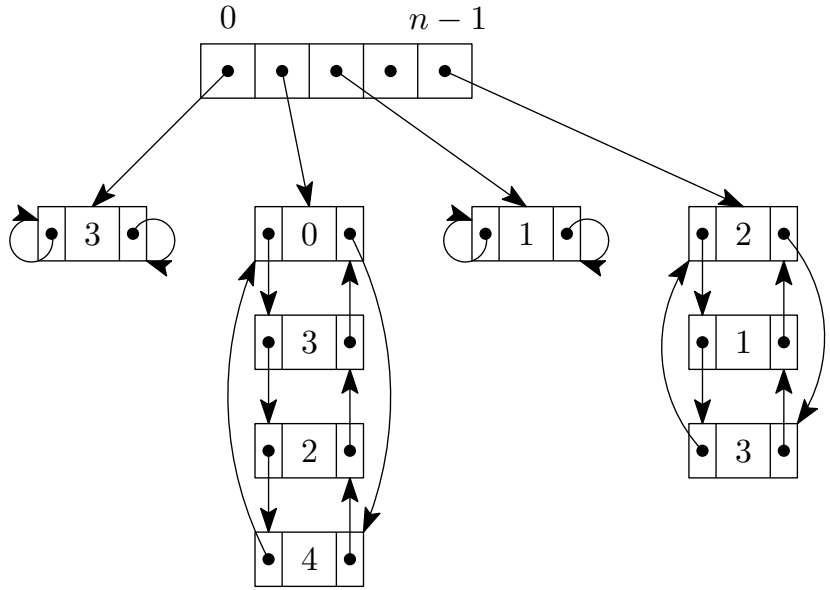




Λίστες γειτνίασης

γράφημα $G(V, E)$ όπου $|V| = n$ και $|E| = m$





Χαρακτηριστικά

- κατανάλωση χώρου $n + 3m$ λέξεις
- επιπλέον πληροφορίες μπορούν να αποθηκευθούν μέσα στις λίστες ή στον πίνακα των κόμβων
- δυναμική αναπαράσταση - εύκολη προσθήκη/αφαίρεση ακμών
- πλοήγηση (π.χ εκτύπωση εξερχόμενων ακμών) χρόνο ίσο με τον βαθμό του κόμβου $\mathcal{O}(d)$
- ερώτημα ακμής $\mathcal{O}(d)$ χρόνο

Γραφήματα με βάρη

Όταν χρησιμοποιούμε γραφήματα για να απαντήσουμε ερωτήματα διαδρομών σε μία συσκευή GPS, θέλουμε να βρούμε την συντομότερη διαδρομή μεταξύ δύο σημείων:

Γραφήματα με βάρη

Όταν χρησιμοποιούμε γραφήματα για να απαντήσουμε ερωτήματα διαδρομών σε μία συσκευή GPS, θέλουμε να βρούμε την συντομότερη διαδρομή μεταξύ δύο σημείων:

- έχουμε υπολογίσει από πριν μία προσέγγιση του μέσου χρόνου που χρειάζεται για να διανύσουμε ένα κομμάτι δρόμου
- αντιστοιχούμε σε κάθε ακμή στο γράφημα ένα αριθμό που υποδηλώνει τον χρόνο που χρειάζεται για να διανύσουμε το κομμάτι του δρόμου στο οποίο αυτή η ακμή αντιστοιχεί

Γενικά, μπορούμε να αντιστοιχίσουμε πληροφορία με τους κόμβους αλλά και τις ακμές ενός γραφήματος. Ένα άλλο παράδειγμα είναι η αντιστοιχία ονόματος με κάθε ακμή.

Αναπαράσταση Γραφημάτων

Άσκηση

Θέλετε να αποθηκεύσετε ένα γράφημα που αναπαριστά το οδικό δίκτυο της πόλης που ζείτε. Απαντήστε στα παρακάτω ερωτήματα:

1. Τι θα αναπαριστά ο κάθε κόμβος του γραφήματος;
2. Τι θα αναπαριστά η κάθε ακμή του γραφήματος;
3. Τι επιπλέον πληροφορία θα πρέπει να υπάρχει στις ακμές και στους κόμβους ώστε να μπορείτε να απαντάτε σε ερωτήματα συντομότερης διαδρομής αλλά και να μπορείτε να ζωγραφίσετε τον χάρτη;
4. Πια αναπαράσταση από τις προηγούμενες είναι η καταλληλότερη; Γιατί; Εξηγήστε.

Αναζήτηση κατά πλάτος (BFS)

Breadth-first search

Η αναζήτηση κατά πλάτος είναι ένας από τους πιο απλούς αλγόριθμους αναζήτησης στα γραφήματα και χρησιμοποιείται ως βάση για πολλούς σημαντικούς αλγόριθμους γραφημάτων.

Είσοδος

- γράφημα $G(V, E)$ και
- ένας αρχικός κόμβος s

Λειτουργία

Ο αλγόριθμος BFS συστηματικά εξερευνά τις ακμές του G ώστε να ανακαλύψει όλους τους κόμβους που είναι προσβάσιμοι από τον κόμβο s.

Αναζήτηση κατά πλάτος (BFS)
Έξοδος

Αποστάσεις Την απόσταση (ελάχιστο αριθμό ακμών) από τον s σε όλους τους προσβάσιμους κόμβους.

Κατα-πλάτος Δέντρο Ένα "κατά-πλάτος δέντρο" (breadth-first tree) με ρίζα τον s που περιέχει όλους τους προσβάσιμους κόμβους.

Για κάθε κόμβο v προσβάσιμο από τον s, το μονοπάτι στο BFS δέντρο αντιστοιχεί σε ελάχιστο μονοπάτι στο γράφημα.

Αναζήτηση κατά πλάτος (BFS)

Κατευθυνόμενα και μη-κατευθυνόμενα

Ο αλγόριθμος λειτουργεί και σε κατευθυνόμενα και σε μη-κατευθυνόμενα γραφήματα.

Αναζήτηση κατά πλάτος (BFS)

Κατευθυνόμενα και μη-κατευθυνόμενα

Ο αλγόριθμος λειτουργεί και σε κατευθυνόμενα και σε μη-κατευθυνόμενα γραφήματα.

Μέτωπο αναζήτησης

Ονομάζεται αναζήτηση κατά πλάτος γιατί μεγαλώνει το μέτωπο της αναζήτησης σε όλο το πλάτος του.

Με άλλα λόγια, ο αλγόριθμος ανακαλύπτει όλους τους κόμβους σε απόσταση k από τον s πριν ανακαλύψει κάποιον κόμβο σε απόσταση $k + 1$.

Αναζήτηση κατά πλάτος (BFS)

Για να μετρά την πρόοδο, ο αλγόριθμος BFS χρωματίζει τους κόμβους του γραφήματος με τρία χρώματα α) λευκό, β) γκρί, και γ) μαύρο.

- Όλοι οι κόμβοι αρχίζουν λευκοί και στην συνέχεια μπορούν να γίνουν γκρί και μετά μαύροι.
- Ένας κόμβος *ανακαλύπτεται* την πρώτη φορά που εμφανίζεται στην αναζήτηση, και γίνεται μη-λευκός.
- Οι γκρί και μαύροι κόμβοι έχουν ανακαλυφθεί.
- Οι γκρί κόμβοι αναπαριστούν το μέτωπο της αναζήτησης και μπορεί να έχουν λευκούς γείτονες.

Αναζήτηση κατά πλάτος (BFS)

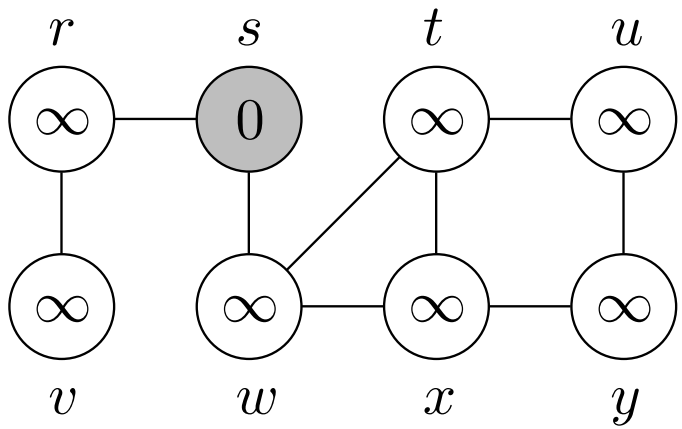
Ψευδοκώδικας

BFS(G, s)

for *κάθε κόμβο* $u \in V \setminus \{s\}$ **do**
    $color[u] = \text{ΛΕΥΚΟ}$
    $d[u] = \infty$
    $\pi[u] = nil$
end
$color[s] = \text{ΓΚΡΙ}$
$d[s] = 0$
$\pi[s] = nil$
$Q = \{s\}$
while $Q \neq \emptyset$ **do**
    $u = Q.head()$
    for *κάθε γείτονα* v *του* u **do**
        if $color[v] = \Lambda EYKO$ **then**
            $color[v] = \text{ΓΚΡΙ}$
            $d[v] = d[u] + 1$
            $\pi[v] = u$
            $Q.enqueue(v)$
        end
    end
    Q.dequeue()
    $color[u] = \text{ΜΑΥΡΟ}$
end

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα

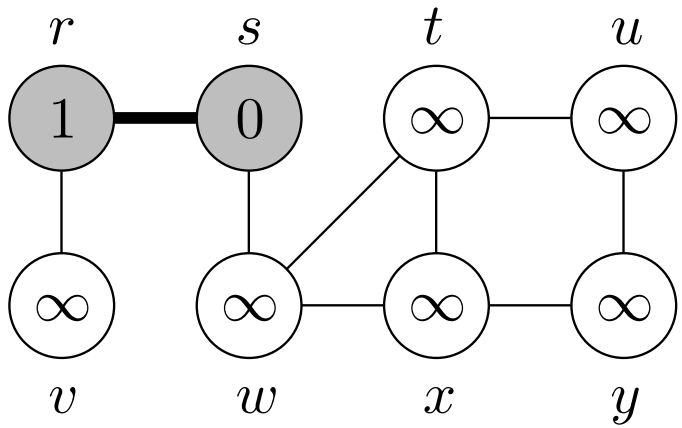


Q $\boxed{s}$

0

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα

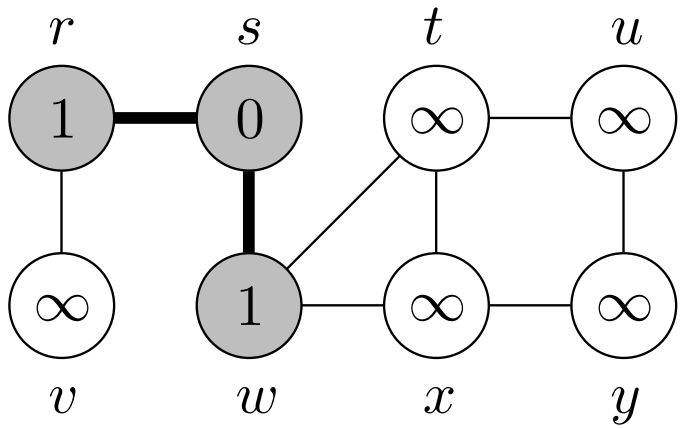


Q $\boxed{r}$

1

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα

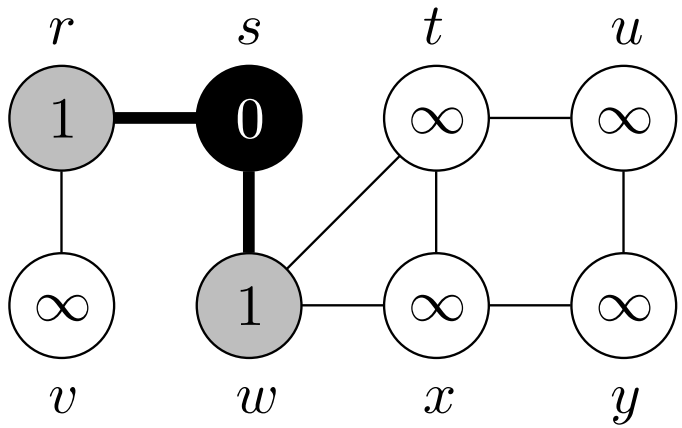


Q | r | w

1 1

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα

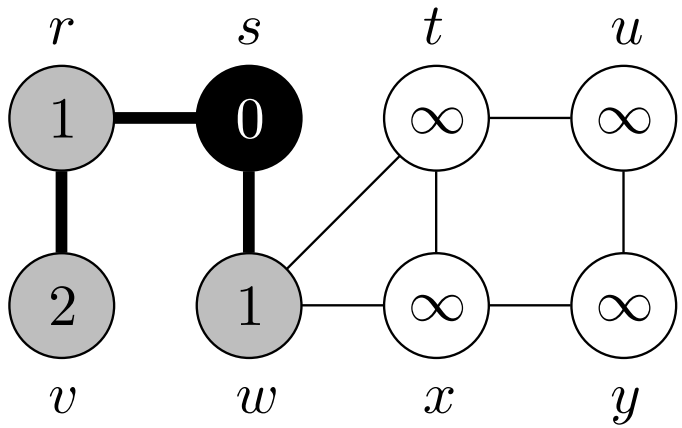


Q | r | w

1 1

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα

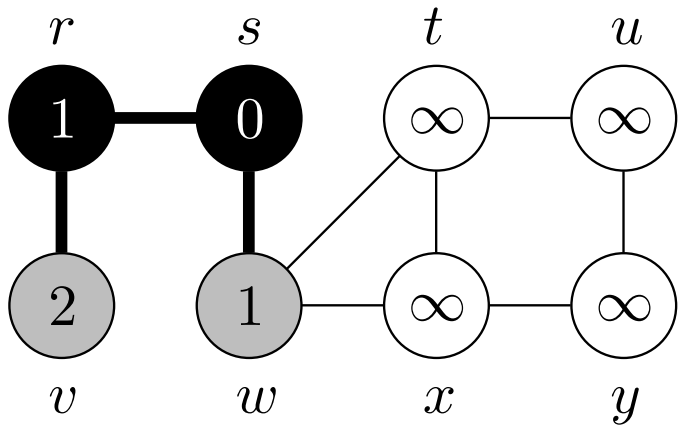


Q $\boxed{w}$ $\boxed{v}$

1 2

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα

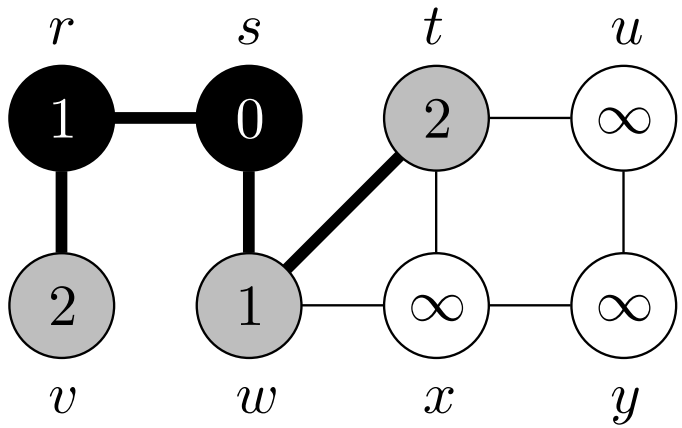


Q $\boxed{w}$ $\boxed{v}$

1 2

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα

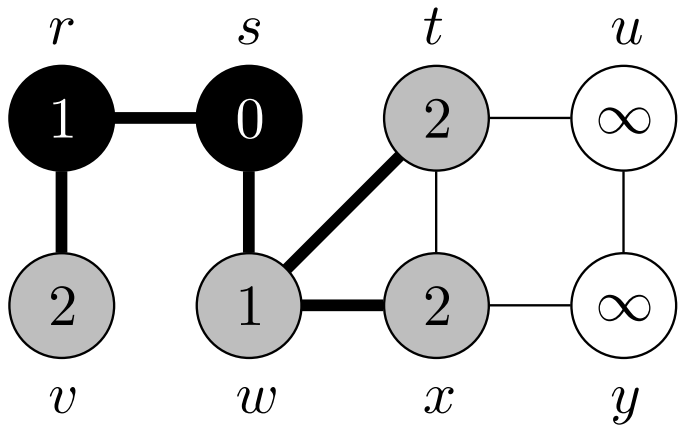


Q | v | t

2 2

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα

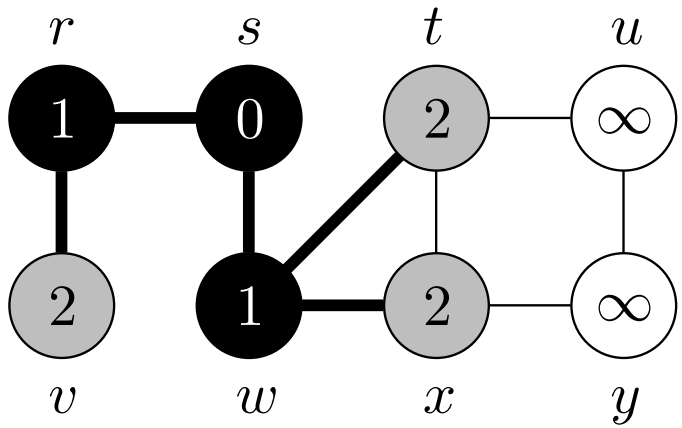


Q $\boxed{v}$ $\boxed{t}$ $\boxed{x}$

2 2 2

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα

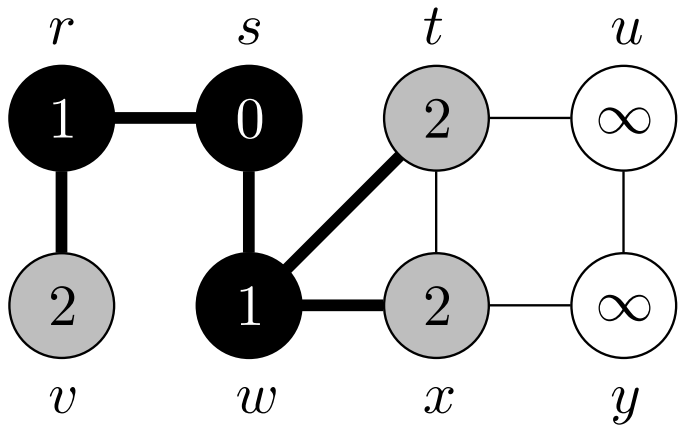


Q | v | t | x

2 2 2

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα

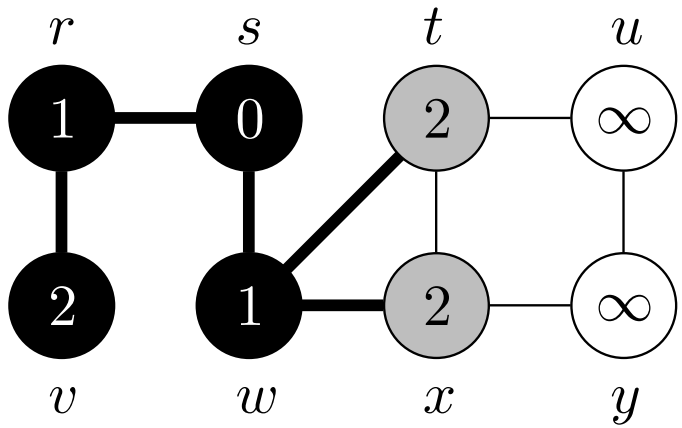


Q $\boxed{t}$ $\boxed{x}$

2 2

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα

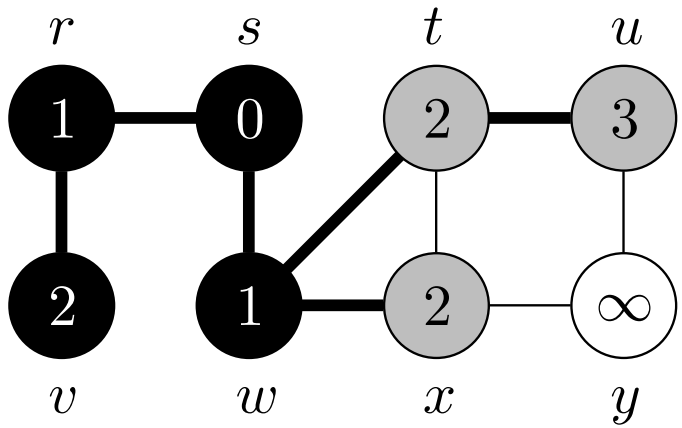


Q $\boxed{t}$ $\boxed{x}$

2 2

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα

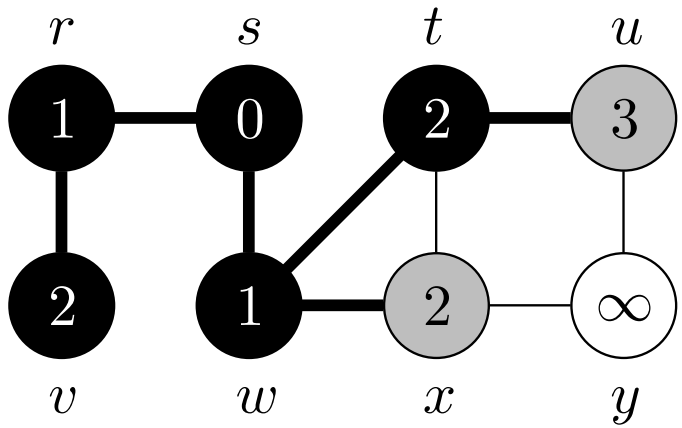


Q $\boxed{x}$ $\boxed{u}$

2 3

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα

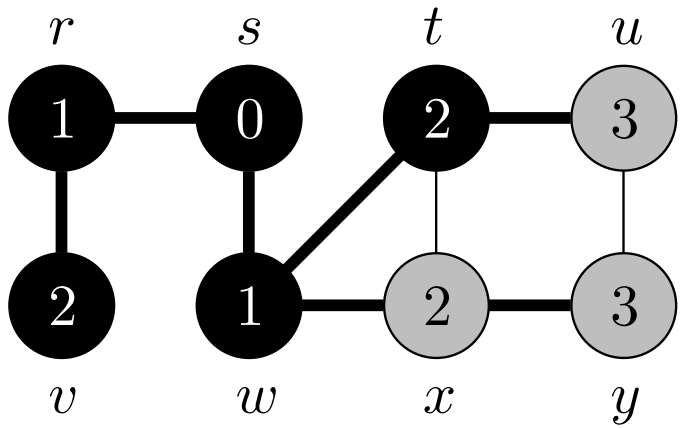


Q $\boxed{x}$ $\boxed{u}$

2 3

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα

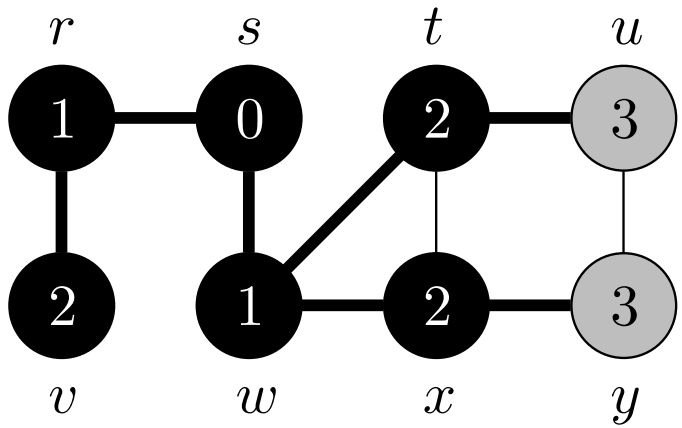


Q $\boxed{u}$ $\boxed{y}$

3 3

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα

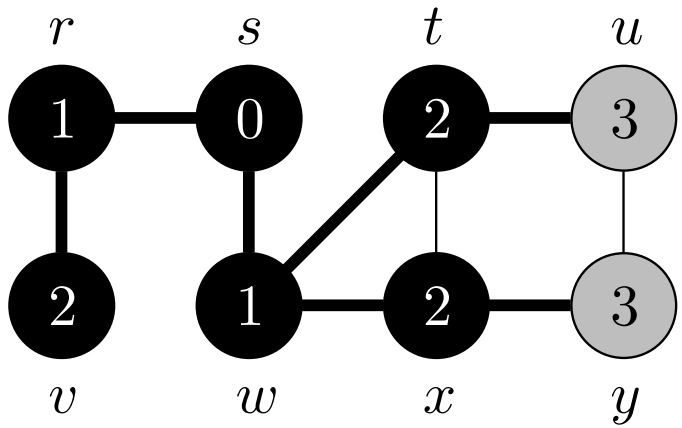


Q $\boxed{u}$ $\boxed{y}$

3 3

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα

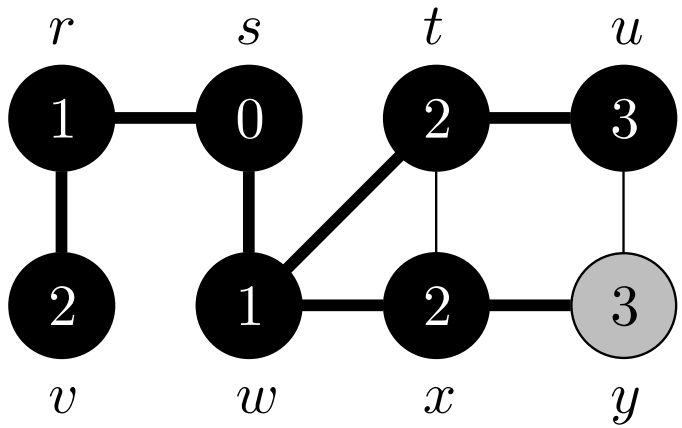


Q $\boxed{y}$

3

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα

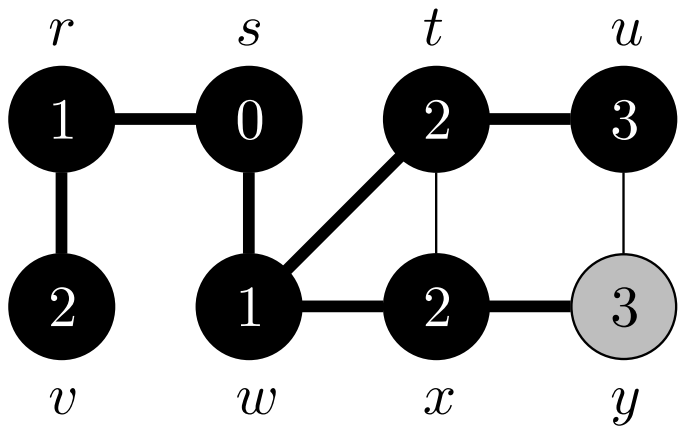


Q $\boxed{y}$

3

Αναζήτηση κατά πλάτος (BFS)

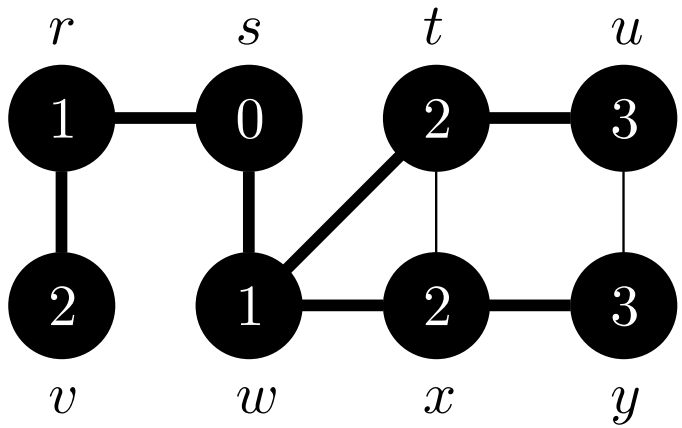
Παράδειγμα



Q □

Αναζήτηση κατά πλάτος (BFS)

Παράδειγμα



Q □

Αναζήτηση κατά πλάτος (BFS)

Χρόνος Εκτέλεσης

Εαν υποθέσουμε αναπαράσταση γραφήματος με λίστες γειτνίασης.

Αναζήτηση κατά πλάτος (BFS)

Χρόνος Εκτέλεσης

Εαν υποθέσουμε αναπαράσταση γραφήματος με λίστες γειτνίασης.

Αρχικοποίηση

$\mathcal{O}(n)$ αφού αρχικοποιούμε πίνακες που έχουν ένα στοιχείο ανά κόμβο.

Αναζήτηση κατά πλάτος (BFS)

Χρόνος Εκτέλεσης

Εαν υποθέσουμε αναπαράσταση γραφήματος με λίστες γειτνίασης.

Αρχικοποίηση

$\mathcal{O}(n)$ αφού αρχικοποιούμε πίνακες που έχουν ένα στοιχείο ανά κόμβο.

Κόστος Λειτουργιών Ουράς

Μετά την αρχικοποίηση κανένας κόμβος δεν γίνεται λευκός και άρα κάθε κόμβος εισάγεται στην ουρά το πολύ μία φορά. Άρα αφαιρείται το πολύ μία φορά.

Η ουρά χρειάζεται $\mathcal{O}(1)$ για τις λειτουργίες αυτές και άρα $\mathcal{O}(n)$ συνολικά.

Αναζήτηση κατά πλάτος (BFS)

Χρόνος Εκτέλεσης

Εαν υποθέσουμε αναπαράσταση γραφήματος με λίστες γειτνίασης.

Αρχικοποίηση

$\mathcal{O}(n)$ αφού αρχικοποιούμε πίνακες που έχουν ένα στοιχείο ανά κόμβο.

Κόστος Λειτουργιών Ουράς

Μετά την αρχικοποίηση κανένας κόμβος δεν γίνεται λευκός και άρα κάθε κόμβος εισάγεται στην ουρά το πολύ μία φορά. Άρα αφαιρείται το πολύ μία φορά.

Η ουρά χρειάζεται $\mathcal{O}(1)$ για τις λειτουργίες αυτές και άρα $\mathcal{O}(n)$ συνολικά.

Κόστος Διάσχισης Ακμών

Η λίστα των ακμών κάθε κόμβου διασχίζεται μόνο μία φορά όταν ο κόμβος αφαιρείται από την ουρά. Το άθροισμα του μεγέθους των λιστών γειτνίασης όλων των κόμβων είναι $O(m)$.

Αναζήτηση κατά πλάτος (BFS)

Χρόνος Εκτέλεσης

Εαν υποθέσουμε αναπαράσταση γραφήματος με λίστες γειτνίασης.

Αρχικοποίηση

$\mathcal{O}(n)$ αφού αρχικοποιούμε πίνακες που έχουν ένα στοιχείο ανά κόμβο.

Κόστος Λειτουργιών Ουράς

Μετά την αρχικοποίηση κανένας κόμβος δεν γίνεται λευκός και άρα κάθε κόμβος εισάγεται στην ουρά το πολύ μία φορά. Άρα αφαιρείται το πολύ μία φορά.

Η ουρά χρειάζεται $\mathcal{O}(1)$ για τις λειτουργίες αυτές και άρα $\mathcal{O}(n)$ συνολικά.

Κόστος Διάσχισης Ακμών

Η λίστα των ακμών κάθε κόμβου διασχίζεται μόνο μία φορά όταν ο κόμβος αφαιρείται από την ουρά. Το άθροισμα του μεγέθους των λιστών γειτνίασης όλων των κόμβων είναι $O(m)$.

Συνολικός χρόνος εκτέλεσης $O(n+m)$.

Αναζήτηση κατά βάθος (DFS)

Depth-first search

Η αναζήτηση κατά βάθος προσπαθεί να πάει όσο πιο βαθιά στο γράφημα γίνεται.

Είσοδος

- γράφημα $G(V, E)$ και
- ένας αρχικός κόμβος s

Λειτουργία

Ξεκινά από τον κόμβο s και ακολουθεί μία μία όλες τις εξερχόμενες ακμές του s εκτελώντας την ίδια λειτουργία αναδρομικά.

Εαν σε κάποιο κόμβο δεν έχει άλλη διέξοδο, επιστρέφει στον κόμβο από όπου προήλθε και συνεχίζει να ακολουθεί ανεξερεύνητες ακμές.

Αναζήτηση κατά βάθος (DFS)

Για να μετρά την πρόοδο, ο αλγόριθμος χρωματίζει τους κόμβους του γραφήματος με τρία χρώματα α) λευκό, β) γκρί, και γ) μαύρο.

- Όλοι οι κόμβοι αρχίζουν λευκοί και στην συνέχεια μπορούν να γίνουν γκρί και μετά μαύροι.
- Όταν ένας κόμβος ανακαλυφθεί γίνεται γκρί και παραμένει γκρί μέχρι να εξερευνηθούν αναδρομικά όλοι οι γείτονες τους. Στην συνέχεια γίνεται μαύρος.
- Ο αλγόριθμος καταγράφει δύο χρόνους για κάθε κόμβο, τον χρόνο που γίνεται γκρί και άρα ανακαλύπτεται για πρώτη φορά και τον χρόνο τερματισμού δηλαδή τον χρόνο που γίνεται μαύρος.

Αναζήτηση κατά βάθος (DFS)

Ψευδοκώδικας

---

dfs(G, s)

---

for *κάθε κόμβο* $u \in V$ **do**
    $color[u] = \Lambda\text{ΕΥΚΟ}$
    $\pi[u] = nil$
end
$time = 1$
dfsvisit(G, s);

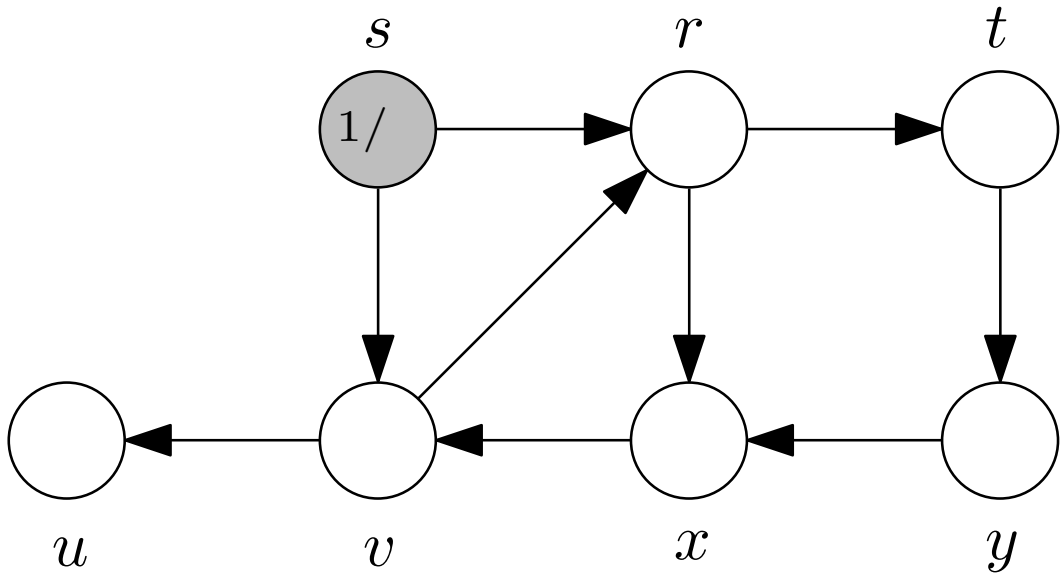
---

dfsvisit(G, u)

$color[u] = \Gamma\text{ΚΡΙ}$
$d[u] = time$
$time = time + 1$
for *κάθε γείτονα v του u* **do**
    if $color[v] = \Lambda EYKO$ **then**
        $\pi[v] = u$
        dfsvisit(G, v)
    end
end
$color[u] = \text{ΜΑΥΡΟ}$
$f[u] = time$
$time = time + 1$

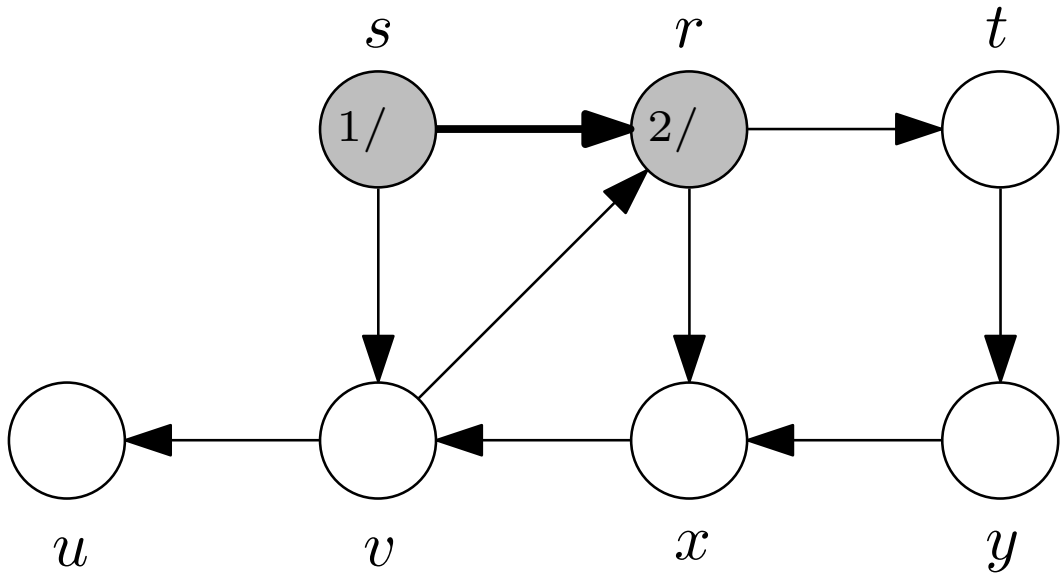
Διάσχιση Γραφημάτων

Depth-first search (DFS)



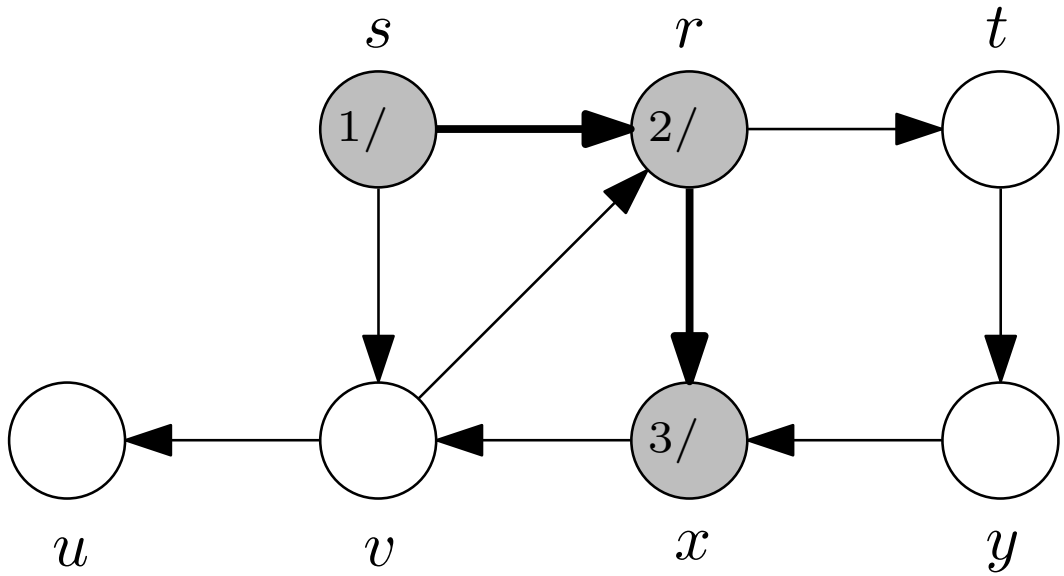
Διάσχιση Γραφημάτων

Depth-first search (DFS)



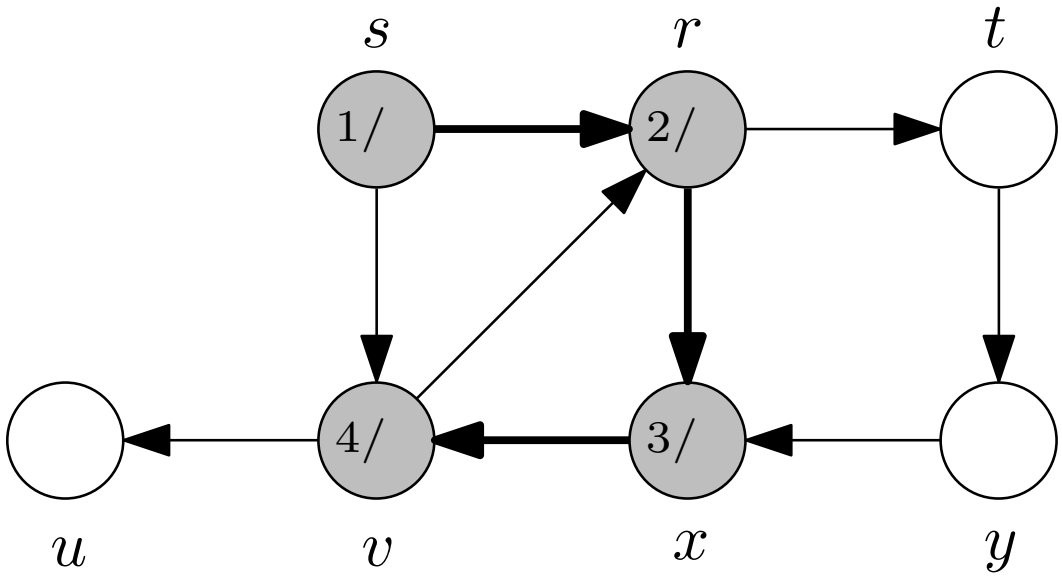
Διάσχιση Γραφημάτων

Depth-first search (DFS)



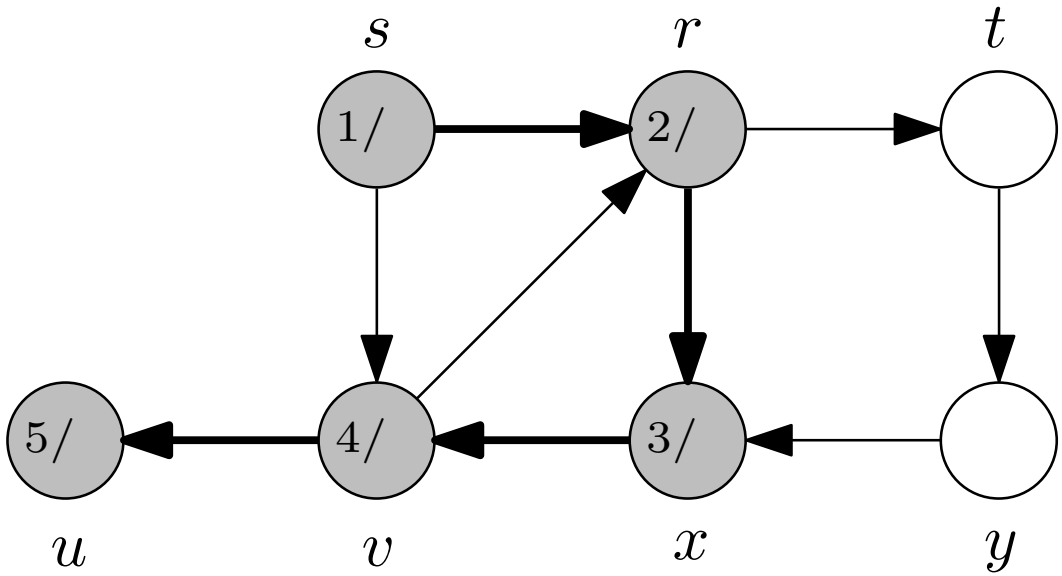
Διάσχιση Γραφημάτων

Depth-first search (DFS)



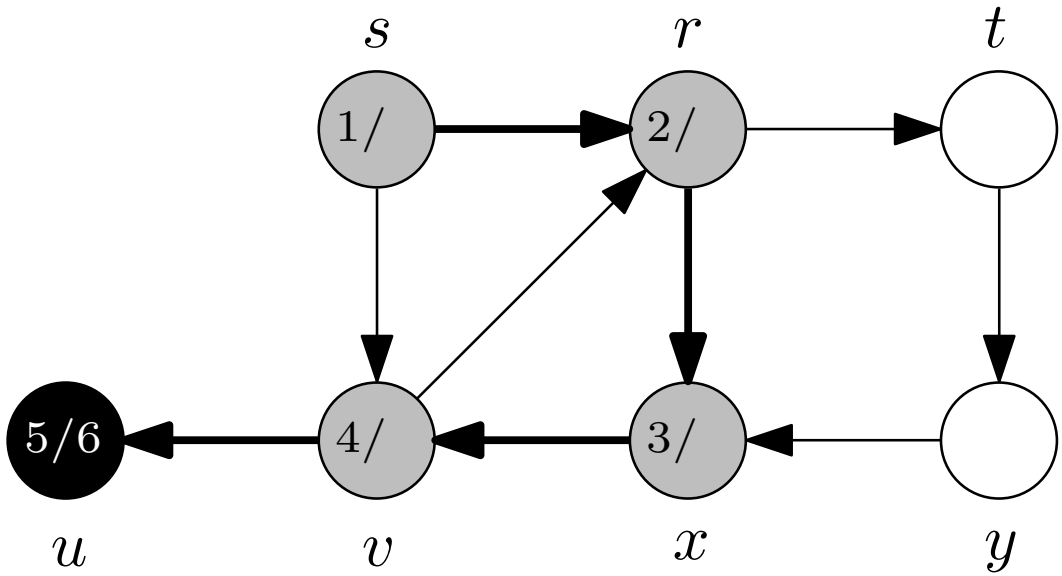
Διάσχιση Γραφημάτων

Depth-first search (DFS)



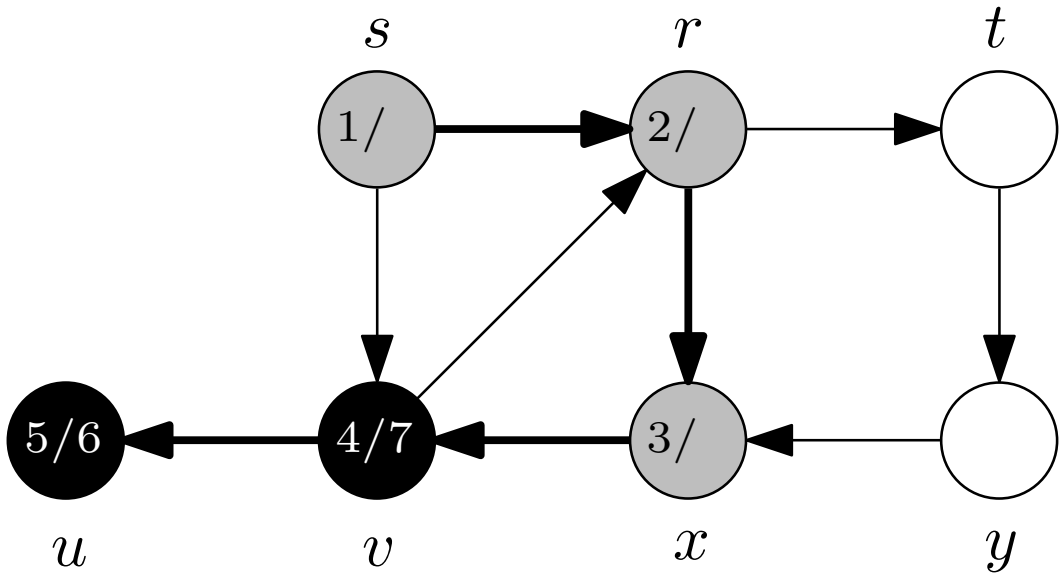
Διάσχιση Γραφημάτων

Depth-first search (DFS)



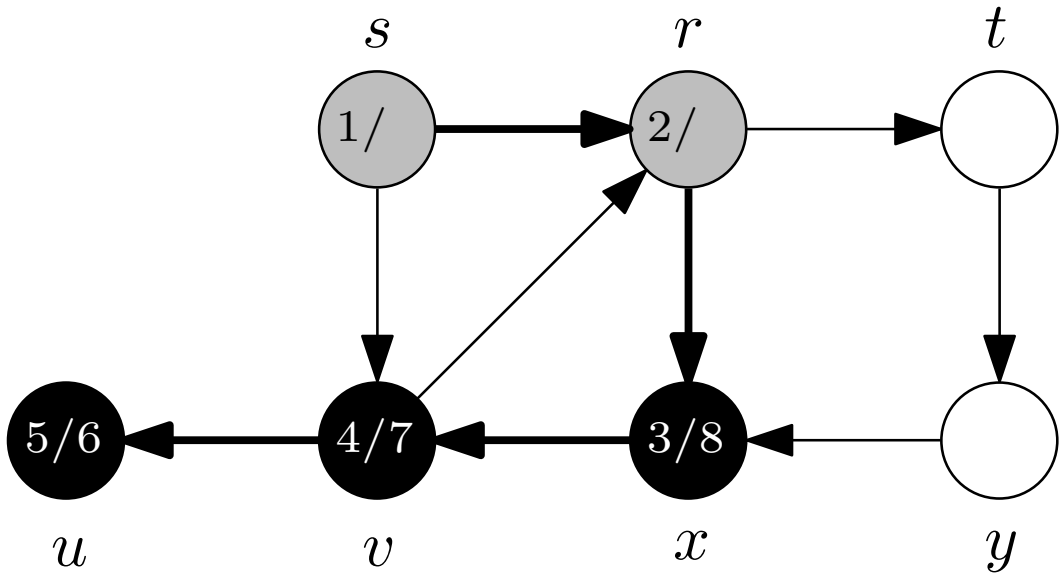
Διάσχιση Γραφημάτων

Depth-first search (DFS)



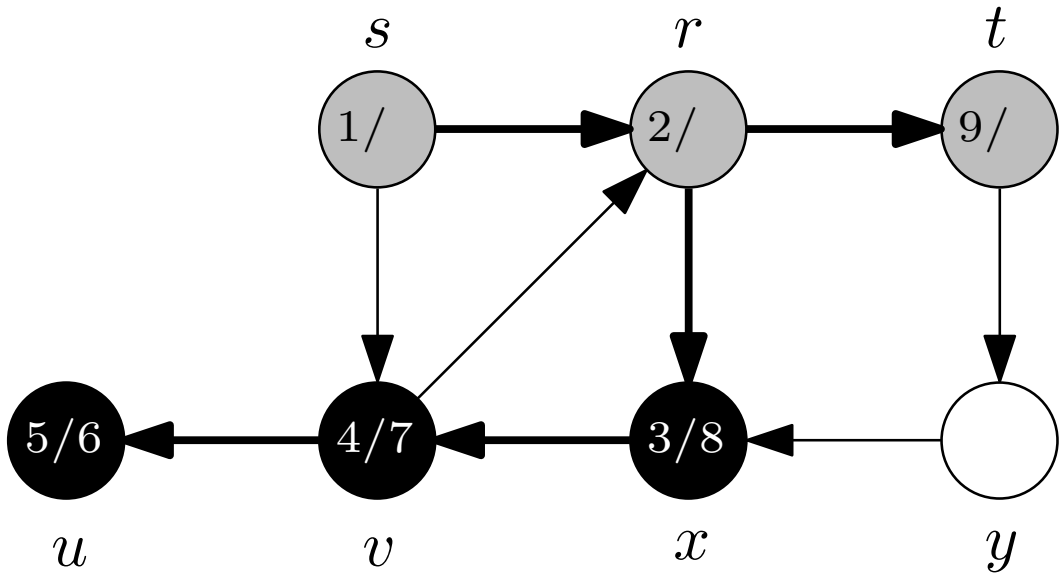
Διάσχιση Γραφημάτων

Depth-first search (DFS)



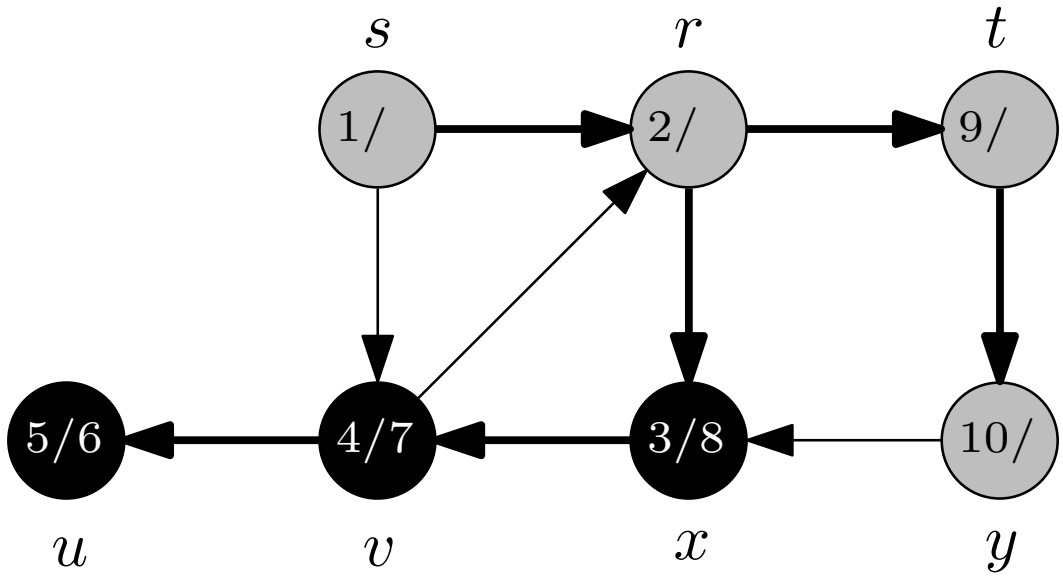
Διάσχιση Γραφημάτων

Depth-first search (DFS)



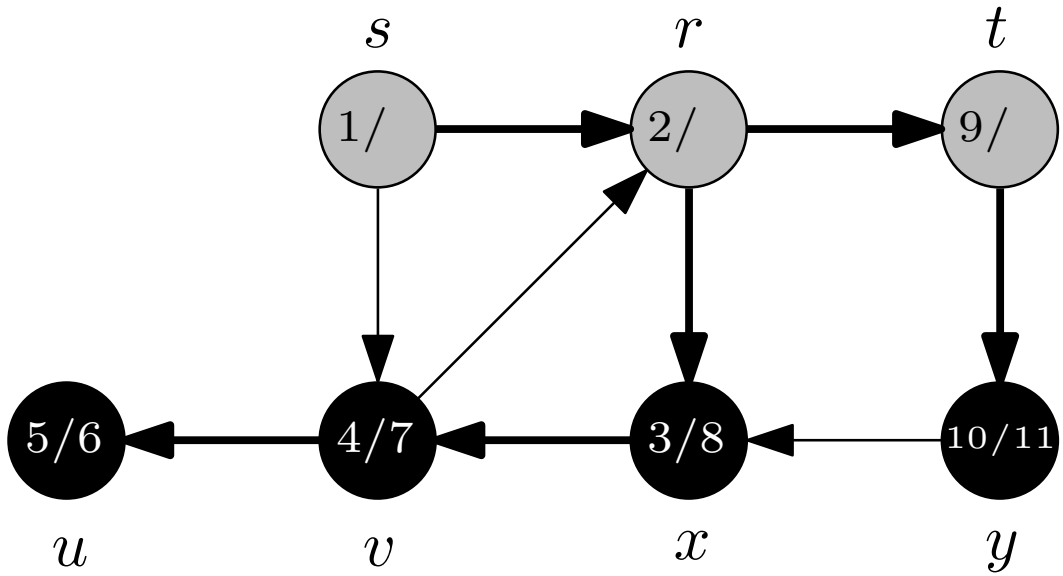
Διάσχιση Γραφημάτων

Depth-first search (DFS)



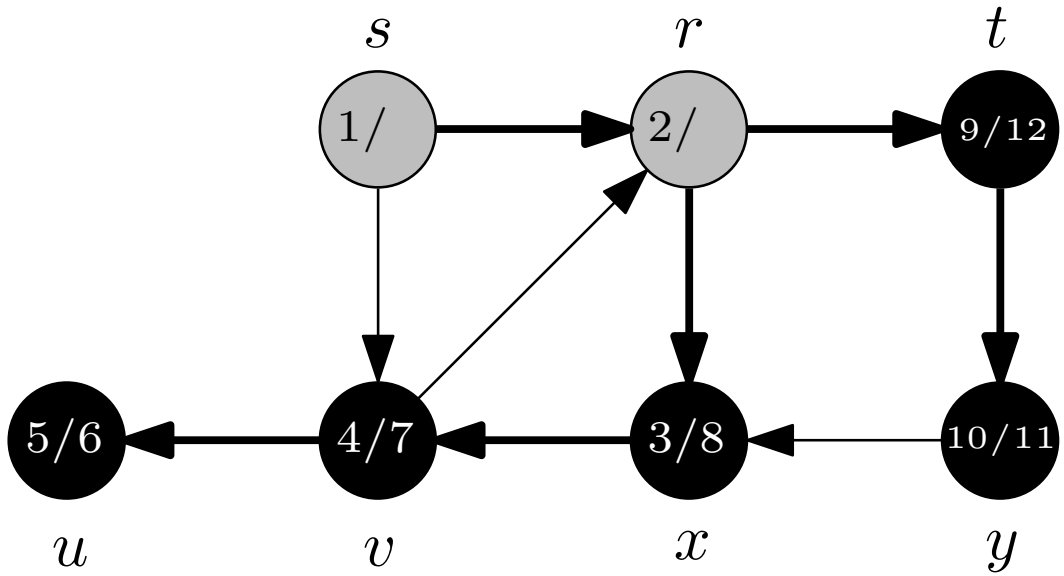
Διάσχιση Γραφημάτων

Depth-first search (DFS)



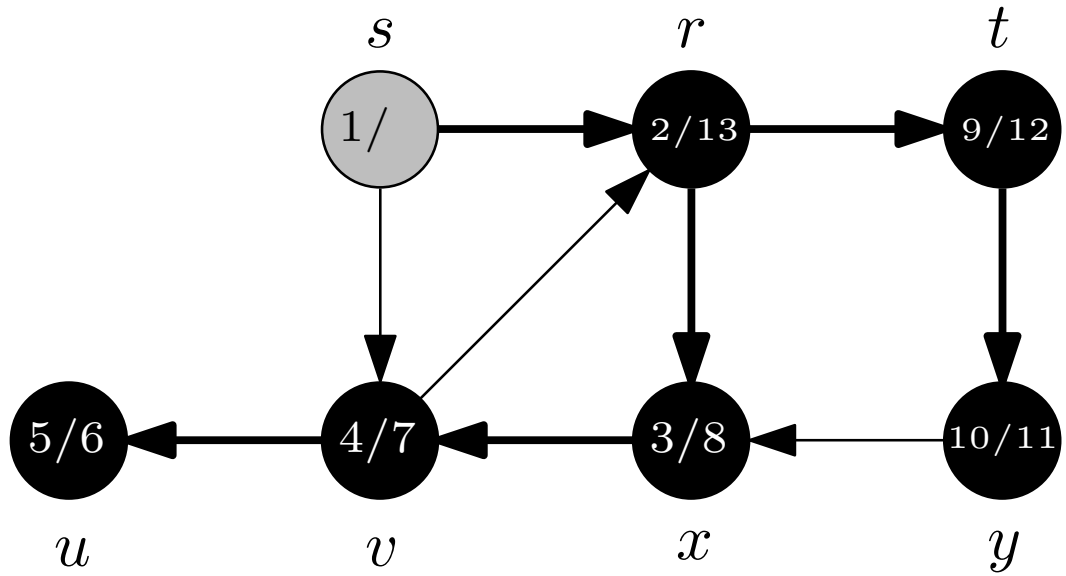
Διάσχιση Γραφημάτων

Depth-first search (DFS)



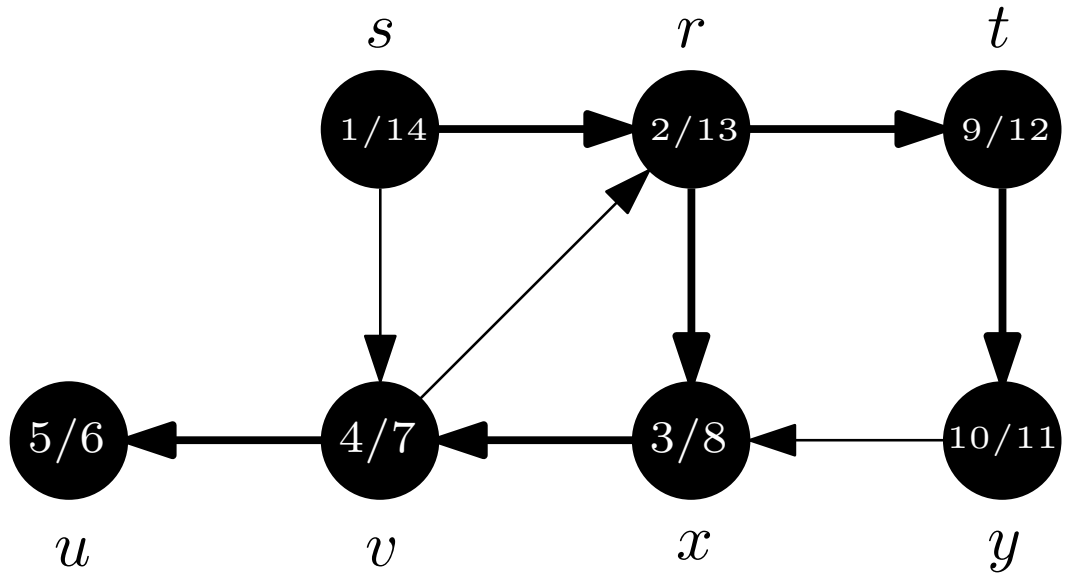
Διάσχιση Γραφημάτων

Depth-first search (DFS)



Διάσχιση Γραφημάτων

Depth-first search (DFS)



Αναζήτηση κατά βάθος (DFS)

Χρόνος Εκτέλεσης

Εαν υποθέσουμε αναπαράσταση γραφήματος με λίστες γειτνίασης.

Αναζήτηση κατά βάθος (DFS)

Χρόνος Εκτέλεσης

Εαν υποθέσουμε αναπαράσταση γραφήματος με λίστες γειτνίασης.

Αρχικοποίηση

$\mathcal{O}(n)$ αφού αρχικοποιούμε πίνακες που έχουν ένα στοιχείο ανά κόμβο.

Αναζήτηση κατά βάθος (DFS)

Χρόνος Εκτέλεσης

Εαν υποθέσουμε αναπαράσταση γραφήματος με λίστες γειτνίασης.

Αρχικοποίηση

$\mathcal{O}(n)$ αφού αρχικοποιούμε πίνακες που έχουν ένα στοιχείο ανά κόμβο.

Κλήσεις της *dfsvisit*(G, v)

Η αναδρομική συνάρτηση *dfsvisit*(G, v) καλείται μία φορά σε κάθε κόμβο αφού καλείται μόνο σε λευκούς κόμβους και το πρώτο πράγμα που κάνει είναι να αλλάξει το χρώμα του κόμβου σε γκρί.

Μία κλήση της *dfsvisit*(G, v) κοστίζει όσο το μήκος της λίστας γειτνίασης του κόμβου v. Το άθροισμα όλων των λιστών γειτνίασης είναι $\mathcal{O}(m)$.

Εαν υποθέσουμε αναπαράσταση γραφήματος με λίστες γειτνίασης.

Αρχικοποίηση

$\mathcal{O}(n)$ αφού αρχικοποιούμε πίνακες που έχουν ένα στοιχείο ανά κόμβο.

Κλήσεις της *dfsvisit*(G, v)

Η αναδρομική συνάρτηση *dfsvisit*(G, v) καλείται μία φορά σε κάθε κόμβο αφού καλείται μόνο σε λευκούς κόμβους και το πρώτο πράγμα που κάνει είναι να αλλάξει το χρώμα του κόμβου σε γκρί.

Μία κλήση της *dfsvisit*(G, v) κοστίζει όσο το μήκος της λίστας γειτνίασης του κόμβου v. Το άθροισμα όλων των λιστών γειτνίασης είναι $\mathcal{O}(m)$.

Συνολικός χρόνος εκτέλεσης $O(n + m)$.

Διάβασμα και Επιπλέον Πηγές

- Ενότητες 8.1, 8.2, 8.3, 8.4, 9.1 και 9.2

  Kurt Mehlhorn και Peter Sanders. Αλγόριθμοι και Δομές Δεδομένων, Τα βασικά εργαλεία, Έκδοση 1η, Κλειδάριθμος, 2014.
- Ενότητες 3.7 και 5.8

  Robert Sedgewick. Αλγόριθμοι σε C: Θεμελιώδεις Έννοιες, Δομές Δεδομένων, Ταξινόμηση, Αναζήτηση. Έκδοση 3η, Κλειδάριθμος, 2006.
- Ενότητες 3.1, 3.2, 3.3, 4.1, 4.2

  Sanjoy Dasgupta, Christos Papadimitriou και Umesh Vazirani, Αλγόριθμοι, Εκδόσεις Κλειδάριθμος, 2008
- Ενότητες 22.1, 22.2, 22.3

  Cormen, Leiserson, Rivest και Stein, Εισαγωγή στους Αλγορίθμους (σε ένα τόμο). Έκδοση 1η, Πανεπιστημιακές Εκδόσεις Κρήτης, 2012.
- Κεφάλαιο 15

  Γεώργιος Φρ. Γεωργακόπουλος, Δομές Δεδομένων: Έννοιες, Τεχνικές, Αλγόριθμοι, Πανεπιστημιακές Εκδόσεις Κρήτης, Ηράκλειο 2002.