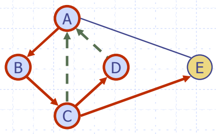
Depth-First Search

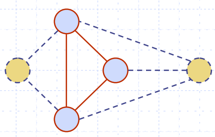


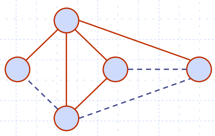
Περιγραφή και Υλικό Ανάγνωσης

- Ορισμοί (§6.1)
  - Υπογράφοι
  - Συνδεσιμότητα
  - Spanning δέντρα και δάση (forests)
- Depth-first search (§6.3.1)
  - Αλγόριθμος
  - Παράδειγμα
  - Ιδιότητες
  - Ανάλυση
- Εφαρμογές του DFS (§6.5)
  - Εύρεση μονοπατιών
  - Εύρεση κύκλων

Υπογράφοι

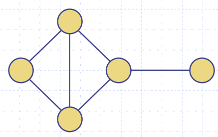
- Ένας υπογράφος S ενός γράφου G είναι ένας γράφος έτσι ώστε
  - Οι κορυφές του S είναι υποσύνολο των κορυφών του G
  - Οι ακμές του S είναι υποσύνολο των ακμών του G
- Ένας spanning υπογράφος του G είναι ένας υπογράφος που περιέχει όλες τις κορυφές του G

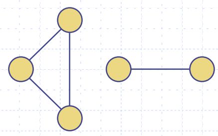

Subgraph


Spanning subgraph

Συνδεσιμότητα

- Ένας γράφος είναι συνδεδεμένος αν υπάρχει ένα μονοπάτι ανάμεσα σε κάθε ζευγάρι κορυφών
- Ένα συνδετικό στοιχείο (connected component) ενός γράφου G είναι ένας μέγιστος συνδεδεμένος υπογράφος του G

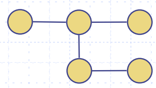

Connected graph

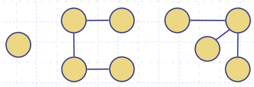

Non connected graph with two connected components

Δέντρα και Δάση

- Ένα (ελεύθερο) δέντρο είναι ένας μη κατευθυνόμενος γράφος T έτσι ώστε
  - Ο T είναι συνδεδεμένος
  - Ο T δεν έχει κύκλους

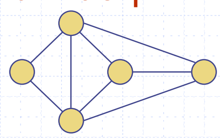
  Αυτός ο ορισμός του δέντρου είναι διαφορετικός από αυτόν του δέντρου με ρίζα
- Ένα δάσος είναι ένας μη κατευθυνόμενος γράφος χωρίς κύκλους
- Τα συνδετικά στοιχεία ενός δάσους είναι δέντρα

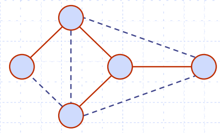

Tree


Forest

Spanning Δέντρα και Δάση

- Ένα spanning δέντρο ενός συνδεδεμένου γράφου είναι ένας spanning υπογράφος ο οποίος είναι δέντρο
- Ένα spanning δέντρο δεν είναι μοναδικό εκτός αν ο γράφος είναι δέντρο
- Τα spanning δέντρα έχουν εφαρμογές στη σχεδίαση δικτύων επικοινωνιών
- Ένα spanning δάσος ενός γράφου είναι ένας spanning υπογράφος ο οποίος είναι δάσος


Graph


Spanning tree

Depth-First Search

- Ο Depth-first search (DFS) είναι μια γενική τεχνική για τη διάσχιση ενός γράφου
- Μια διάσχιση DFS ενός γράφου G
  - Επισκέπτεται όλες τις ακμές και τις κορυφές του G
  - Ορίζει αν ο G είναι συνδεδεμένος
  - Υπολογίζει τα συνεκτικά στοιχεία του G
  - Υπολογίζει ένα spanning δάσος του G
- Ο DFS σε ένα γράφο με n κορυφές και m ακμές παίρνει $O(n + m)$ χρόνο
- Ο DFS μπορεί να επεκταθεί για να λύσει και άλλα προβλήματα γράφων
  - Βρες και ανάφερε ένα μονοπάτι μεταξύ δύο δοσμένων κορυφών
  - Βρες ένα κύκλο στο γράφο
- Ο Depth-first search είναι για τους γράφους ότι το Euler tour για τα δυαδικά δέντρα

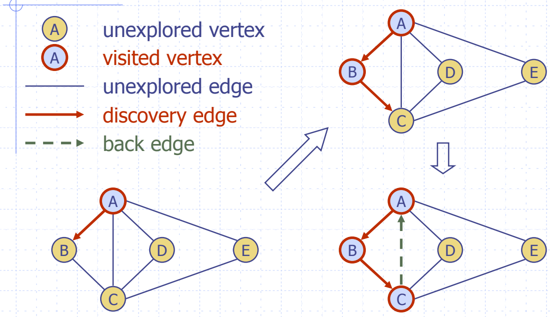
DFS Algorithm

- Ο αλγόριθμος χρησιμοποιεί ένα μηχανισμό για να θέτει και να παίρνει "labels" από κορυφές και ακμές

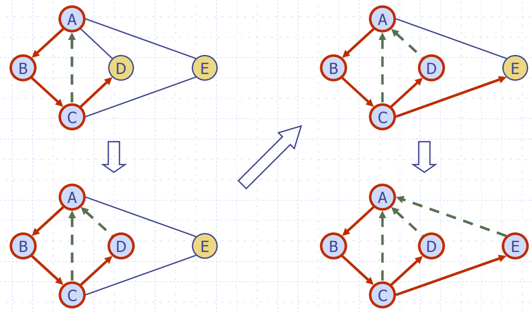
Algorithm ***DFS(G)***
  Input graph *G*
  Output labeling of the edges of *G*
    as discovery edges and
    back edges
  for all $u \in$ *G.vertices*()
    setLabel(*u, UNEXPLORED*)
  for all $e \in$ *G.edges*()
    setLabel(*e, UNEXPLORED*)
  for all $v \in$ *G.vertices*()
    if *getLabel*(*v*) = *UNEXPLORED*
      DFS(*G, v*)

Algorithm ***DFS(G, v)***
  Input graph *G* and a start vertex *v* of *G*
  Output labeling of the edges of *G*
    in the connected component of *v*
    as discovery edges and back edges
  setLabel(*v, VISITED*)
  for all $e \in$ *G.incidentEdges*(*v*)
    if *getLabel*(*e*) = *UNEXPLORED*
      $w \leftarrow$ *opposite*(*v,e*)
      if *getLabel*(*w*) = *UNEXPLORED*
        setLabel(*e, DISCOVERY*)
        DFS(*G, w*)
      else
        setLabel(*e, BACK*)

Παράδειγμα

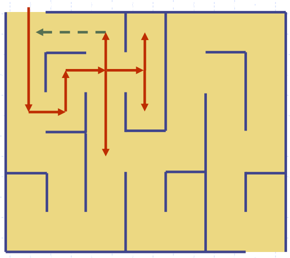


Παράδειγμα (συν.)



DFS and Maze Traversal

- Ο αλγόριθμος DFS είναι παρόμοιος με την κλασσική στρατηγική για την εξερεύνηση ενός λαβυρίνθου
  - Σημειώνουμε κάθε διασταύρωση, γωνία και αδιέξοδο (vertex) που επισκεπτόμαστε
  - Σημειώνουμε κάθε διαδρομή (edge) που διασχίζουμε
  - Παρακολουθούμε το μονοπάτι πίσω στην είσοδο (start vertex) με ένα σχοινί (recursion stack)



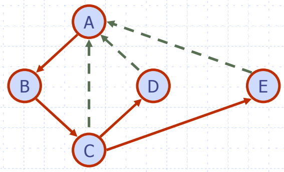
Ιδιότητες του DFS

Ιδιότητα 1

Η $DFS(G, v)$ επισκέπτεται όλες τις κορυφές και τις ακμές του συνεκτικού στοιχείου του v

Ιδιότητα 2

Οι discovery edges που χαρακτηρίστηκαν από τη $DFS(G, v)$ σχηματίζουν ένα spanning δέντρο του συνεκτικού στοιχείου του v



Ανάλυση του DFS

- Η ανάθεση/απόκτηση του label κορυφής/ακμής παίρνει $O(1)$ χρόνο
- Κάθε κορυφή χαρακτηρίζεται δύο φορές
  - Μια ως UNEXPLORED
  - Μια ως VISITED
- Κάθε ακμή χαρακτηρίζεται δύο φορές
  - Μια ως UNEXPLORED
  - Μια ως DISCOVERY ή BACK
- Η μέθοδος incidentEdges καλείται μια φορά για κάθε κορυφή
- Ο DFS τρέχει σε $O(n + m)$ χρόνο δεδομένου ότι ο γράφος αναπαρίσταται από τη δομή λίστας γειτνίασης
  - Υπενθυμίζουμε ότι $\Sigma_v \deg(v) = 2m$

Εύρεση μονοπατιού

- Μπορούμε να ειδικεύσουμε τον αλγόριθμο DFS για να βρει ένα μονοπάτι μεταξύ δύο δοσμένων κορυφών u και z χρησιμοποιώντας την template method pattern
- Καλούμε τη $DFS(G, u)$ με u την αρχική κορυφή
- Χρησιμοποιούμε μια στοίβα S για να παρακολουθούμε το μονοπάτι μεταξύ της αρχικής κορυφής και της τωρινής κορυφής
- Μόλις η κορυφή προορισμού z συναντηθεί, επιστρέφουμε το μονοπάτι ως τα περιεχόμενα της στοίβας

```
Algorithm pathDFS(G, v, z)
  setLabel(v, VISITED)
  S.push(v)
  if v = z
    return S.elements()
  for all e ∈ G.incidentEdges(v)
    if getLabel(e) = UNEXPLORED
      w ← opposite(v,e)
      if getLabel(w) = UNEXPLORED
        setLabel(e, DISCOVERY)
        S.push(e)
        pathDFS(G, w, z)
        S.pop(e)
      else
        setLabel(e, BACK)
  S.pop(v)
```


Εύρεση κύκλων

- Μπορούμε να ειδικεύσουμε τον αλγόριθμο DFS για να βρει ένα απλό κύκλο χρησιμοποιώντας την template method pattern
- Χρησιμοποιούμε μια στοίβα S για να παρακολουθήσουμε το μονοπάτι ανάμεσα στην αρχική κορυφή και την τωρινή κορυφή
- Μόλις μια back edge (v, w) συναντηθεί, επιστρέφουμε τον κύκλο ως το τμήμα της στοίβας από την κορυφή της ως την κορυφή w

```
Algorithm cycleDFS(G, v, z)
  setLabel(v, VISITED)
  S.push(v)
  for all e ∈ G.incidentEdges(v)
    if getLabel(e) = UNEXPLORED
      w ← opposite(v,e)
      S.push(e)
      if getLabel(w) = UNEXPLORED
        setLabel(e, DISCOVERY)
        pathDFS(G, w, z)
        S.pop(e)
      else
        T ← new empty stack
        repeat
          o ← S.pop()
          T.push(o)
        until o = w
        return T.elements()
  S.pop(v)
```