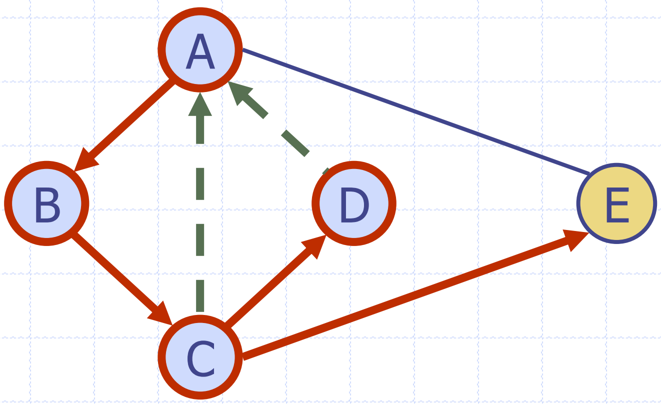
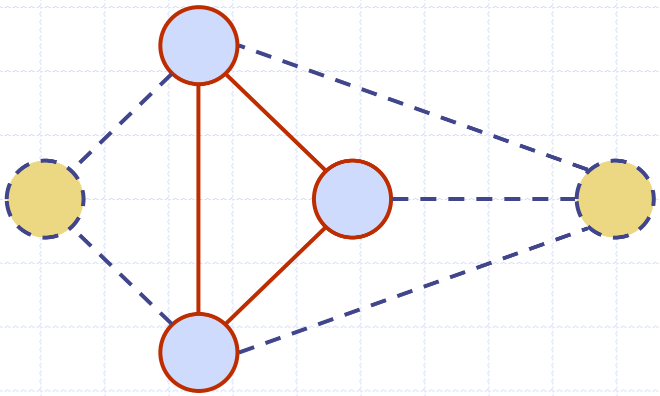
Παρουσίαση για χρήση με το σύγγραμμα, Αλγόριθμοι Σχεδίαση και Εφαρμογές, των M. T. Goodrich and R. Tamassia, Wiley, 2015 (στα ελληνικά από εκδόσεις Μ. Γκιούρδας)

Αναζήτηση πρώτα σε βάθος

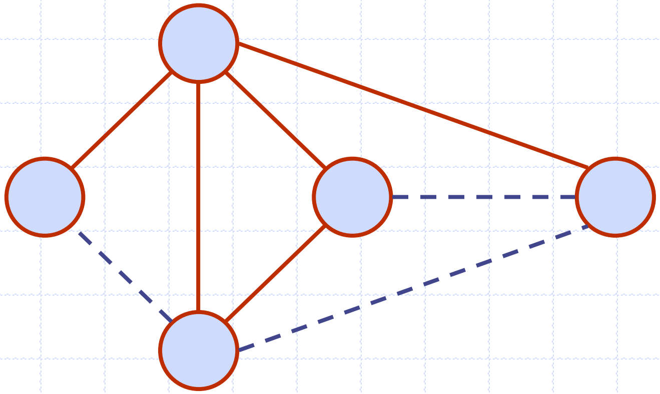


Υπογράφοι

- Ένας υπογράφος **S** ενός γράφου **G** είναι ένας γράφος όπου:
  - Οι κορυφές του **S** είναι υποσύνολο των κορυφών του **G**
  - Οι ακμές του **S** είναι ένα υποσύνολο των ακμών του **G**
- Ένας παράγων υπογράφος (spanning subgraph) του **G** είναι ένας υπογράφος που περιέχει όλες τις κορυφές του **G**



Υπογράφος



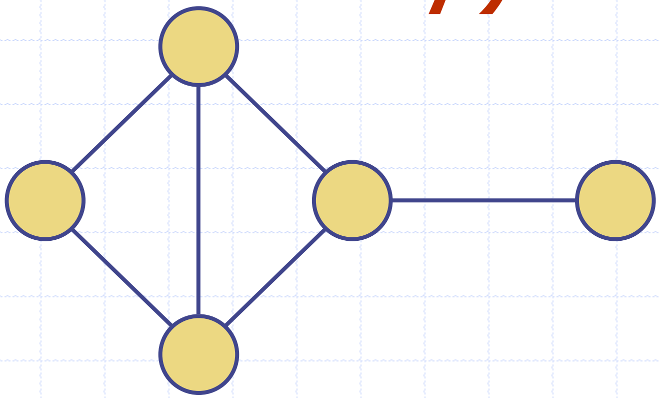
Παράγων υπογράφος

Εφαρμογή: Ανιχνευτής Ιστού

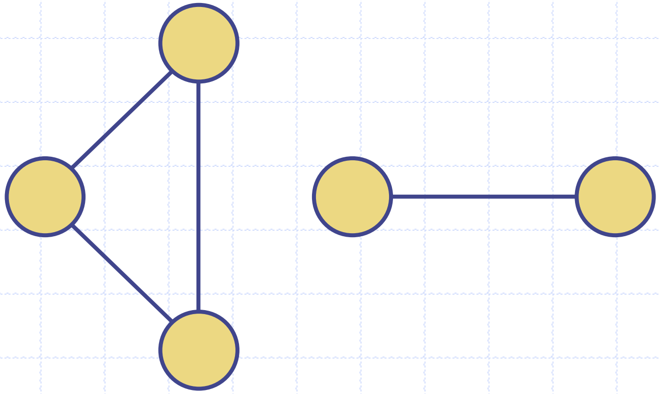
- Μια θεμελιώδης λειτουργία που θα θέλαμε να εκτελέσουμε σε έναν γράφο είναι η **διάσχιση των ακμών και των κορυφών** του συγκεκριμένου γράφου.
- **Διάσχιση (traversal)** είναι η συστηματική διαδικασία εξερεύνησης ενός γράφου μέσω εξέτασης όλων των κορυφών και των ακμών του.
- Για παράδειγμα, τα προγράμματα **web crawler (ανίχνευσης ιστού)**, που συλλέγουν δεδομένα για τις μηχανές αναζήτησης, εξερευνούν γράφους εγγράφων με υπερσυνδέσεις, εξετάζοντας τις κορυφές, που είναι τα έγγραφα και τις ακμές που είναι οι υπερσυνδέσεις μεταξύ εγγράφων
- Μια διάσχιση είναι αποτελεσματική αν επισκέπτεται όλες τις κορυφές και τις ακμές σε γραμμικό χρόνο.

Συνεκτικότητα (connectivity)

- Ένας γράφος είναι συνεκτικός αν για οποιεσδήποτε δύο κορυφές, υπάρχει μία διαδρομή μεταξύ των κορυφών.
- Μία συνεκτική συνιστώσα του γράφου **G** είναι ένας μέγιστος συνεκτικός υπογράφος του **G**



Συνεκτικός γράφος

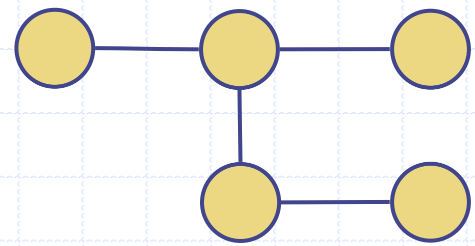


Μη συνεκτικός γράφος με δύο συνεκτικές συνιστώσες

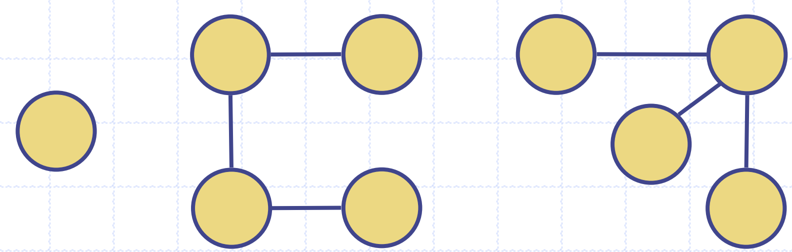
Δέντρα και Δάση

- Ένα (ελεύθερο) δέντρο είναι ένας μη κατευθυνόμενος γράφος T όπου:
  - Ο T είναι συνεκτικός
  - Ο T δεν έχει κύκλους

  Αυτός ο ορισμός του δέντρου διαφέρει από το δέντρο με ρίζα.
- Ένα δάσος είναι ένας μη κατευθυνόμενος γράφος χωρίς κύκλους.
- Οι συνεκτικές συνιστώσες ενός δάσους είναι δέντρα.



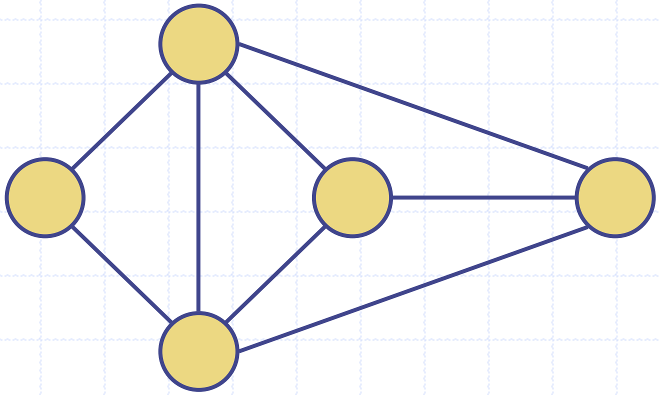
Δέντρο



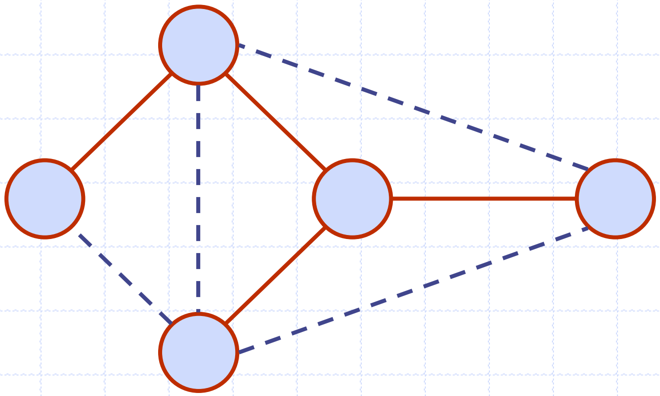
Δάσος

Παράγοντα δέντρα και δάση

- Ένα παράγον δέντρο ενός συνεκτικού γράφου είναι ένας παράγον υπογράφος που είναι δέντρο
- Ένα παράγον δέντρο δεν είναι μοναδικό εκτός αν ο γράφος είναι ένα δέντρο.
- Τα παράγοντα δέντρα βρίσκουν εφαρμογή στα δίκτυα επικοινωνιών.
- Ένα παράγον δάσος ενός γράφου είναι ένας παράγον υπογράφος που είναι ένα δάσος



Γράφος



Παράγον δέντρο

Αναζήτηση πρώτα σε βάθος

- Η αναζήτηση πρώτα σε βάθος (DFS) είναι μία γενική τεχνική διάσχισης γράφου
- Μία DFS διάσχιση ενός γράφου **G**:
  - Επισκέπτεται όλες τις κορυφές και τις ακμές του **G**
  - Διαπιστώνει αν ο **G** είναι συνεκτικός
  - Υπολογίζει τις συνεκτικές συνιστώσες του **G**
  - Υπολογίζει ένα παράγον δάσος του **G**
- Ο DFS σε γράφο με n κορυφές και m ακμές είναι $O(n+m)$ χρονικά
- Ο DFS μπορεί να επεκταθεί ώστε να λύνει κι άλλα προβλήματα γράφων:
  - Εύρεση διαδρομής μεταξύ δύο κορυφών.
  - Εύρεση ενός κύκλου, εάν υπάρχει, σε έναν γράφο.
- Η αναζήτηση πρώτα σε βάθος για τους γράφους είναι ότι είναι η διάσχιση Euler για τα δυαδικά δέντρα.

DFS αλγόριθμος

Algorithm DFS(G, v):

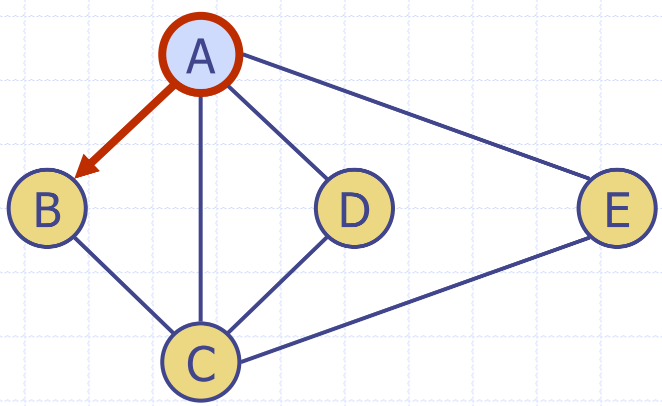
Input: A graph G and a vertex v in G

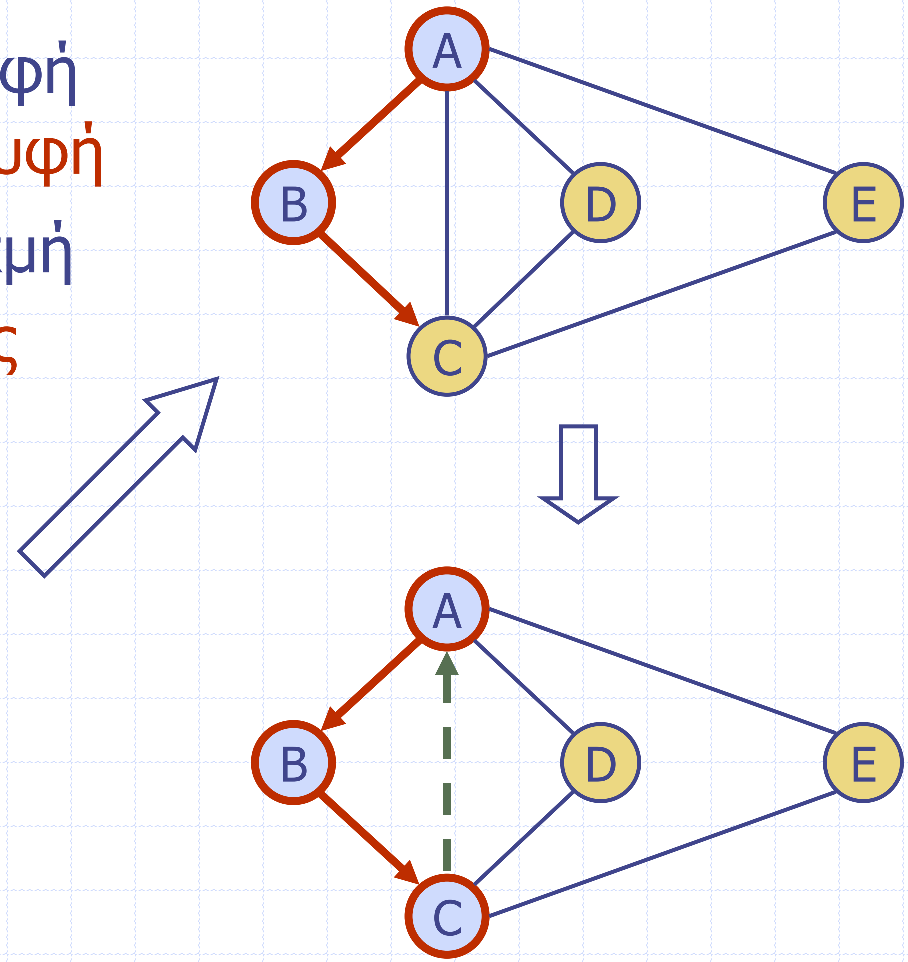
Output: A labeling of the edges in the connected component of v as discovery edges and back edges, and the vertices in the connected component of v as explored

```
Label v as explored
for each edge, e, that is incident to v in G do
    if e is unexplored then
        Let w be the end vertex of e opposite from v
        if w is unexplored then
            Label e as a discovery edge
            DFS(G, w)
        else
            Label e as a back edge
```

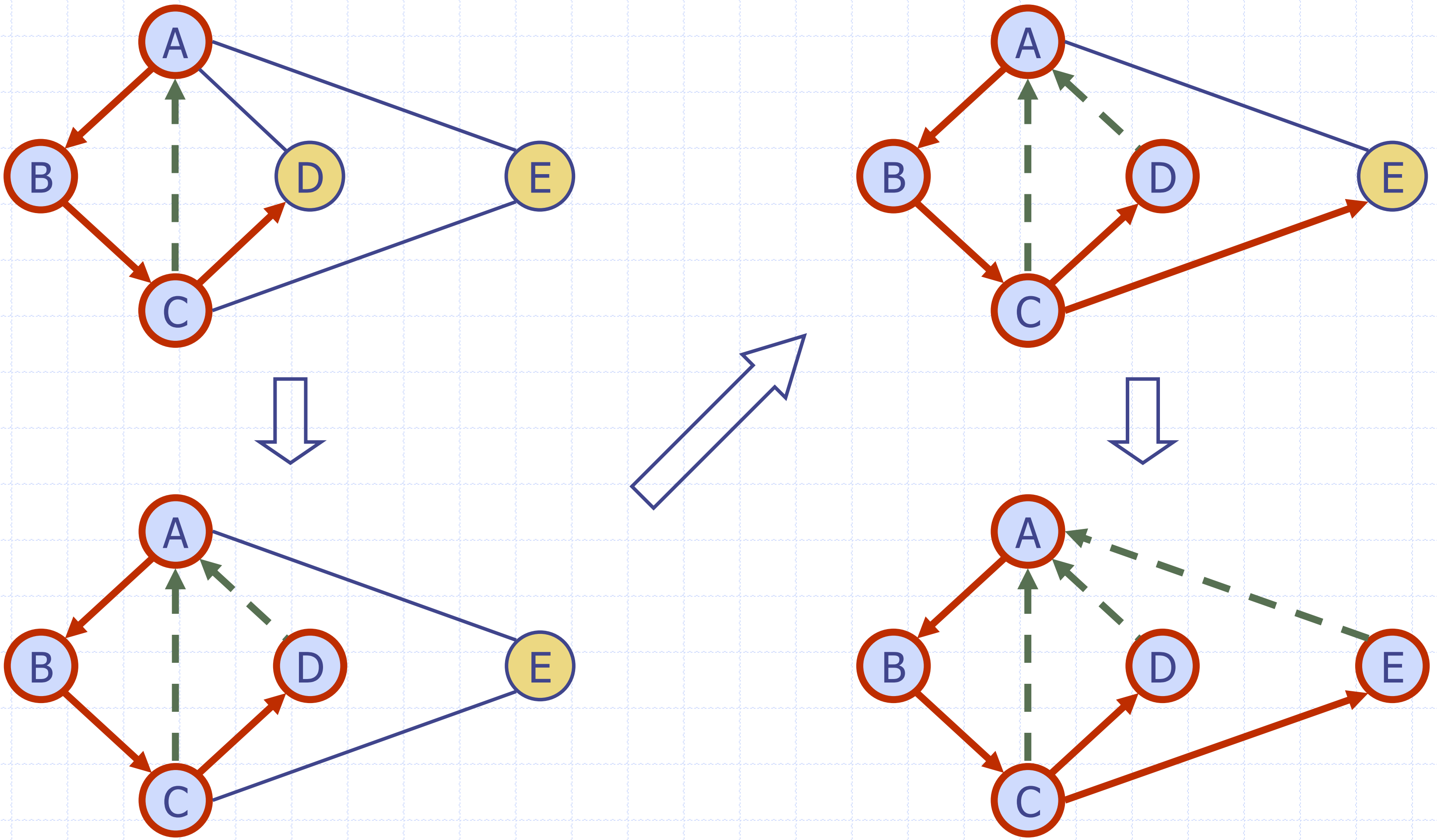
Παράδειγμα

- A Ανεξερεύνητη κορυφή
- A Εξερευνημένη κορυφή
- Ανεξερεύνητη ακμή
- Ακμή εξερεύνησης
- Πίσω ακμή

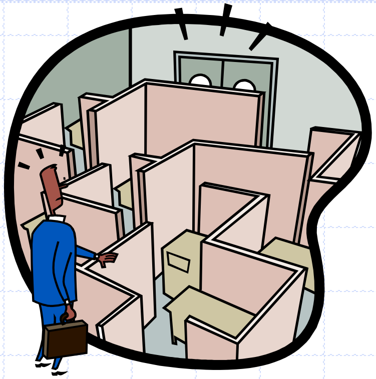




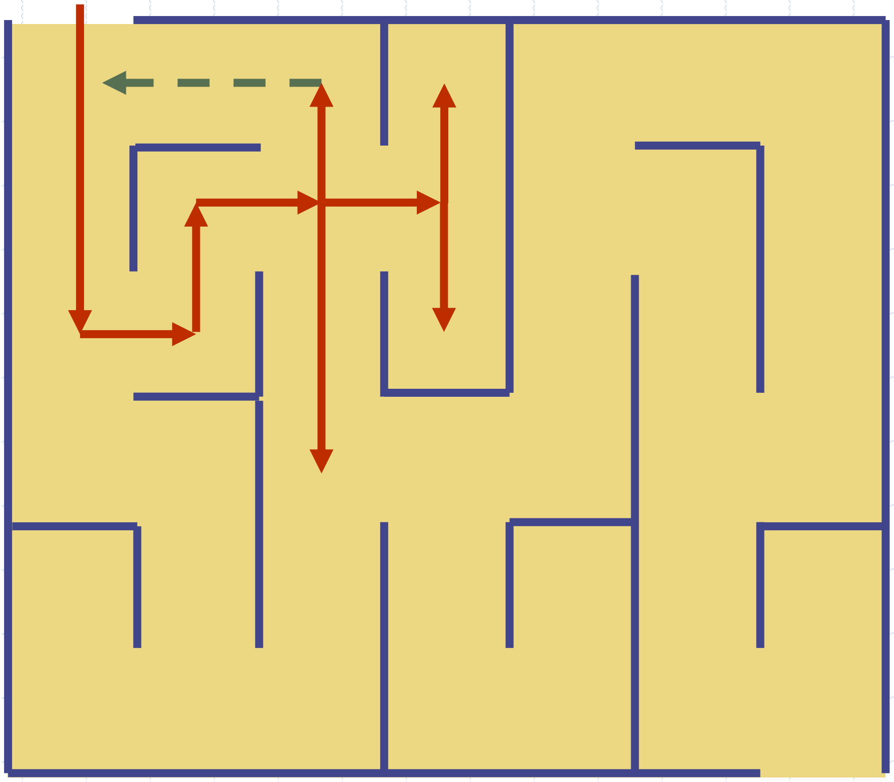
Παράδειγμα (συν.)



DFS και διάσχιση λαβύρινθου



- Ο αλγόριθμος DFS αποτελεί μία κλασική στρατηγική εξερεύνησης λαβυρίνθου.
  - Σημειώνουμε κάθε διασταύρωση, γωνία και αδιέξοδο (κορυφές) που επισκεφτήκαμε
  - Σημειώνουμε κάθε διάδρομο (ακμή) που επισκεφτήκαμε
  - Παρακολουθούμε το μονοπάτι πίσω στην είσοδο (κορυφή εκκίνησης) σαν να διαθέτουμε ένα σχοινί (στοίβα αναδρομής)



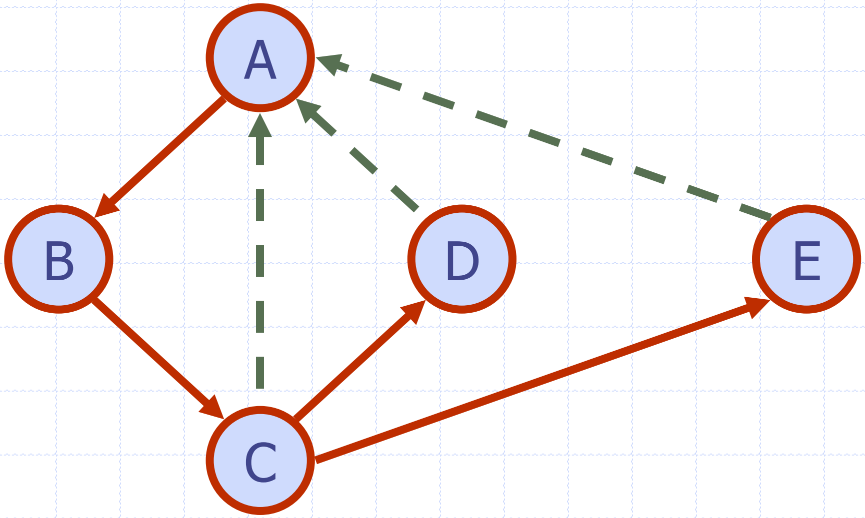
Ιδιότητες του DFS

Ιδιότητα 1

Ο ***DFS*(*G*, *v*)** επισκέπτεται όλες τις κορυφές και ακμές στην συνεκτική συνιστώσα του ***v***

Ιδιότητα 2

Οι ακμές ανακάλυψης του ***DFS*(*G*, *v*)** δημιουργούν έναν παράγον δέντρο της συνεκτικής συνιστώσας του ***v***



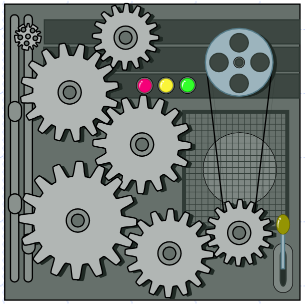
Ο γενικός αλγόριθμος DFS

- Εκτέλεση ενός DFS για κάθε ανεξερεύνητη κορυφή:

Algorithm DFS(G):
    Input: A graph G
    Output: A labeling of the vertices in each connected component of G as explored

    Initially label each vertex in v as unexplored
    for each vertex, v, in G **do**
        if v is unexplored **then**
            DFS(G, v)

Ανάλυση του DFS



- Για τη σημείωση και τη λήψη της σημείωσης μίας κορυφής/ακμής χρειαζόμαστε χρόνο $O(1)$
- Κάθε κορυφή σημειώνεται δύο φορές
  - μια ως ανεξερεύνητη
  - μια ως εξερευνημένη
- Κάθε ακμή σημειώνεται δύο φορές
  - μια ως ανεξερεύνητη
  - μια ως ακμή ανακάλυψης ή πίσω ακμή
- Η μέθοδος incidentEdges καλείται μία φορά για κάθε κορυφή
- Ο DFS εκτελείται σε χρόνο $O(n + m)$ δεδομένου ότι ο γράφος αναπαριστάται από την δομή λίστας γειτνίασης
  - Θυμηθείτε ότι $\Sigma_v \deg(v) = 2m$

Εύρεση διαδρομής

- Μπορούμε να εξειδικεύσουμε τον αλγόριθμο DFS ώστε να βρίσκει μία διαδρομή μεταξύ των κορυφών ***u*** και ***z***
- Καλούμε την ***DFS***(***G***, ***u***) με το ***u*** σαν κορυφή εκκίνησης
- Χρησιμοποιούμε μία στοίβα ***S*** για να παρακολουθούμε την διαδρομή μεταξύ της κορυφής εκκίνησης και της τρέχουσας κορυφής
- Μόλις συναντήσουμε την κορυφή προορισμό ***z***, επιστρέφουμε ως διαδρομή 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)
```

Εντοπισμός κύκλου

- Μπορούμε να χρησιμοποιήσουμε τον αλγόριθμο DFS για να εντοπίζει την ύπαρξη απλού κύκλου στο γράφο.
- Χρησιμοποιούμε μία στοίβα ***S*** για να παρακολουθούμε την διαδρομή μεταξύ της κορυφής εκκίνησης και της τρέχουσας κορυφής
- Όταν συναντήσουμε μια πίσω ακμή **(v, w)** επιστρέφουμε τον κύκλο το κομμάτι της στοίβας από την κορυφή της στοίβας μέχρι την **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)
```