Διάλεξη 30: Τοπολογική Ταξινόμηση

Στην ενότητα αυτή θα μελετηθούν τα εξής επιμέρους θέματα:

- Τοπολογική Ταξινόμηση
- Εφαρμογές, Παραδείγματα, Αλγόριθμοι

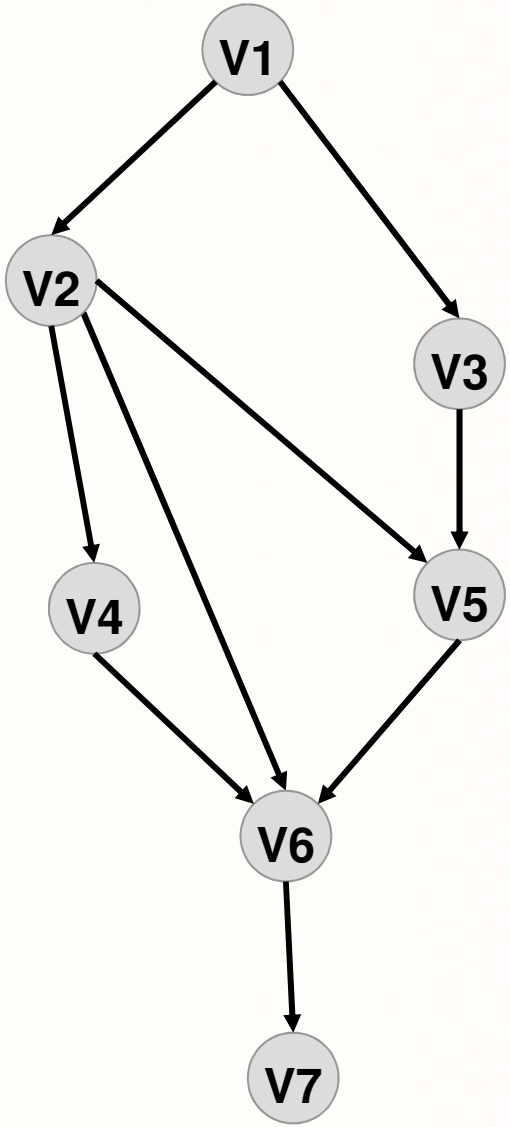
Διδάσκων: Παναγιώτης Ανδρέου

Τοπολογική Ταξινόμηση (Topological Sort)

- Δίνεται ένα **σύνολο εργασιών** και θέλουμε να ορίσουμε τη **σειρά με την οποία πρέπει να εκτελέσει** τις εργασίες ένας επεξεργαστής, δεδομένης της **ύπαρξης περιορισμών** ως προς την προτεραιότητά τους.
- Κάθε εργασία έχει ένα σύνολο προαπαιτούμενων εργασιών, δηλαδή δεν μπορεί να εκτελεσθεί προτού συμπληρωθεί κάθε μια από τις προαπαιτούμενες.
- Μπορούμε να παραστήσουμε το πρόβλημα ως έναν κατευθυνόμενο γράφο:
  - Οι **κορυφές** του γράφου αντιστοιχούν σε κάθε μια από τις εργασίες, και
  - η ύπαρξη **ακμής** από την κορυφή **A** στην κορυφή **B** δηλώνει ότι η εργασία Α πρέπει να εκτελεστεί πριν από τη Β.

- Τοπολογική ταξινόμηση του γράφου είναι **μια σειρά των κορυφών** του, v_1, …, v_n, ώστε αν (v_i, v_j) είναι ακμή του γράφου τότε i εκτελείται πριν το j (δηλαδή i<j).

Παράδειγμα Τοπολογικών Ταξινομήσεων



Τοπολογικές Ταξινομήσεις του γράφου:

1. V1, V2, V4, V3, V5, V6, V7
2. V1, V2, V3, V5, V6, V7, V4
3. V1, V3, V2, V5, V6, V4, V7
4. …

Αλγόριθμος για Τοπολογική Ταξινόμηση

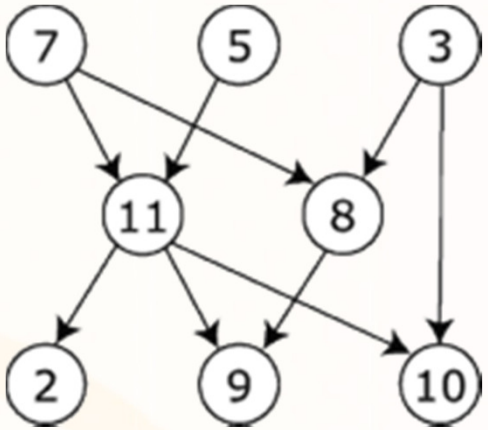
- **<u>Βαθμός εισόδου (in-degree):</u>** ενός κόμβου είναι ο αριθμός των ακμών που καταλήγουν στον κόμβο. (Στο πρόβλημα μας, ο αριθμός των προαπαιτούμενων εργασιών)
- Για κάθε κορυφή **u** έστω **I[u]** ο βαθμός εισόδου (αριθμός γονέων) της **u.**

ΑΛΓΟΡΙΘΜΟΣ

Επαναλαμβάνουμε τα εξής βήματα:

1. διαλέγουμε κορυφή Α με I[A]=0,
2. τυπώνουμε την Α,
3. για όλες τις κορυφές Β, όπου υπάρχει η ακμή (A,B) μειώνουμε την τιμή I[B] κατά 1.

ΠΑΡΑΔΕΙΓΜΑ



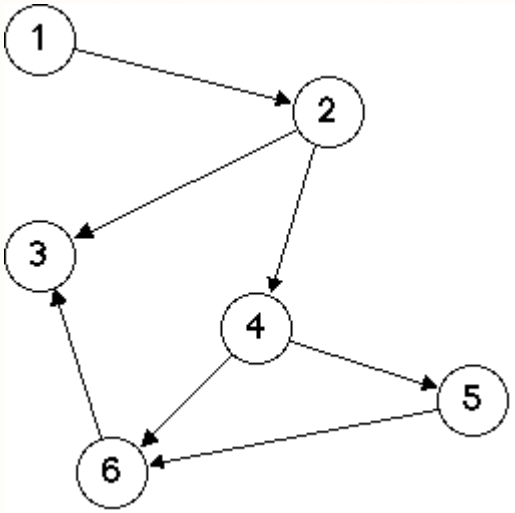
ΑΠΟΤΕΛΕΣΜΑΤΑ(Υπάρχουν αρκετές τοπολογικές ταξινομήσεις)

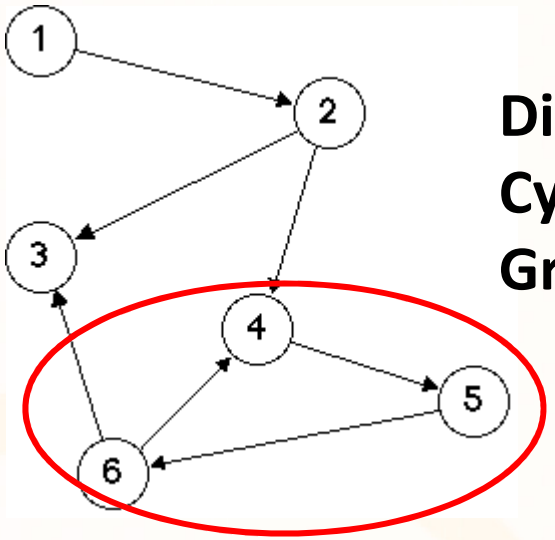
- **7,5,3,11,8,2,9,10**
- **7,5,11,2,3,10,8,9**
- …..

Κατευθυνόμενοι Μη-Κυκλικοί Γράφοι

- Ωστόσο η τοπολογική ταξινόμηση δουλεύει μόνο για μια ειδική κλάση γράφων η οποία ονομάζεται **DAGs**
- **DAG: Directed Acyclic Graphs (Κατευθυνόμενοι Μη-Κυκλικοί Γράφοι)**: Ένας γράφος στο οποίο κανένα μονοπάτι δεν ξεκινά και τελειώνει στον ίδιο κόμβο.

Directed Acyclic Graph





Directed Cyclic Graph

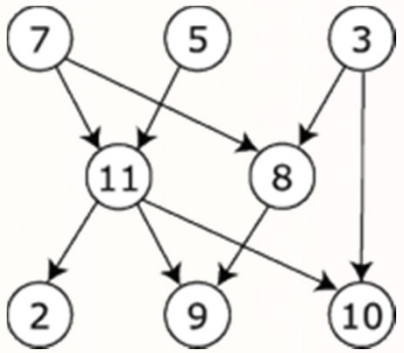
- Το γεγονός ότι δουλεύει μόνο για DAGs είναι αναμενόμενο διότι αν υπήρχαν κυκλικά μονοπάτια τότε δεν θα υπήρχε κάποια σειρά (ordering) μεταξύ των στοιχείων (αφού δεν θα ξέραμε ποια είναι η αρχή)

(Ψευδό)-Υλοποίηση 1

```
topSort1( graph G ){
  // αρχικοποίηση πίνακα μεγέθους |V|
  int I[|V|] = {};
  // μέτρηση in-degree για κάθε κόμβο
  for each vertex u
    for each edge (u,v)                    } O(|V| + |E|)
       I[v]++;
 // προσπέλαση του γράφου
  for (i=1; i <= |V|; i++){
    v = FindVertexOfIndegree0;             } O(|V|)
    if (v == NULL) {// δεν υπάρχει κόμβος
                    //με in-degree=0
          Error("Graph has a cycle");
          return;
    }
    print v;              // εκτύπωση κόμβου
    for each edge (v,w) // μείωση in-degree  } O(|E|)
       I[w]--;          // για κάθε παιδί
  }
```

Χρόνος Εκτέλεσης:

$O(|V|^2+E)$



$O(|V|^2 + |E|)$

(Ψευδό)-Υλοποίηση 2 (Με Χρήση Ουράς)

```
topologicalSort( graph G ){
 Queue Q; // ορισμός βοηθητικής ουράς
 // αρχικοποίηση πίνακα μεγέθους |V|
 int I[|V|] = {};
 // μέτρηση indegree για κάθε κόμβο
 for each vertex u
    for each edge (u,v)
        I[v]++;
 // τοποθέτησε κάθε στοιχείο με indegree=0 σε μια ουρά
 for each vertex u
       if (I[u]==0) Enqueue(u, Q);
 while (! IsEmpty(Q)){
       u = Dequeue(Q);
       output u; number_of_outputs++;
       for each (u,v) {
          I[v]--;
          if (I[v]==0) Enqueue (v, Q);
       }
 }
 // Εάν δεν εκτυπώθηκαν όλοι οι κόμβοι τότε είχε κύκλο α γράφος γιατί
  κάποιοι κόμβοι δεν πήραν ποτέ indegree = 0;
 if (number_of_outputs != |V|)
     Error("Graph has a cycle");
```

Θ(|V| + |E|)

Θ(|V|)

Θ(|V| + |E|)

Χρόνος Εκτέλεσης:

Θ(|V| + |E|)