

# Κατανεμημένα Συστήματα I

## Μάθημα Βασικής Επιλογής, Χειμερινού Εξαμήνου

### Τομέας Εφαρμογών και Θεμελιώσεων

Μάριος Λογαράς, Μαρία Ράπη  
Ιωάννης Χατζηγιαννάκης

Δευτέρα, 14 Νοεμβρίου, 2011  
Αίθουσα Β3



## Προηγούμενο Μάθημα

- ▶ Σύγχρονα Κατανεμημένα Συστήματα
- ▶ Αναζήτηση κατά Εύρος
- ▶ Πρόβλημα Εκλογής Αρχηγού
- ▶ Μελέτη κατανεμημένων αλγόριθμων για Δίκτυα Δακτυλίου
- ▶ Μελέτη ενός κατανεμημένου αλγόριθμου για Γενικά Δίκτυα



## Μοντέλο Σύγχρονου Δικτύου

- ▶ Μία συλλογή υπολογιστικών μονάδων ή `επεξεργαστές`
  - ▶ κάθε επεξεργαστής εκτελεί μόνο μία διεργασία
- ▶ Οι μονάδες του συστήματος είναι συνδεδεμένες με ένα σύγχρονο δίκτυο
  - ▶ Ορίζουμε το σύγχρονο δίκτυο ως ένα **κατευθυνόμενο γράφημα**  $G = (V, E)$
  - ▶ αποτελείται από  $n = |V|$  **κορυφές** και  $m = |E|$  **ακμές**
- ▶ Υποθέτουμε ότι κάθε κανάλι επικοινωνίας μπορεί να δεχτεί μόνο ένα μήνυμα τη φορά
  - ▶ τα κανάλια είναι οι ακμές του γραφήματος
- ▶ Θεωρούμε ότι υπάρχει ένα δεδομένο αλφάβητο  $M$  μηνυμάτων



## Οι Καταστάσεις των Διεργασιών

- ▶ Κάθε διεργασία  $u \in V$  χαρακτηρίζεται από ένα σύνολο καταστάσεων  $states_u$ 
  - ▶ Ορισμένες τις ονομάζουμε **αρχικές καταστάσεις**  $start_u$
  - ▶ Ορισμένες τις ονομάζουμε **καταστάσεις τερματισμού**  $halt_u$
- ▶ Διαθέτει μια γεννήτρια εξερχόμενων μηνυμάτων  $msgs_u : states_u \times nbrs_u^{out} \rightarrow M \cup \{null\}$ 
  - ▶ δεδομένης της τρέχουσας κατάστασης
  - ▶ δημιουργεί κάποια μηνύματα για τις γειτονικές διεργασίες
- ▶ Διαθέτει μία συνάρτηση αλλαγής κατάστασης  $trans_u : states_u \times (M \cup \{null\})^{nbrs_u^{in}} \rightarrow states_u$ 
  - ▶ δεδομένης της τρέχουσας κατάστασης
  - ▶ τα μηνύματα που παραλήφθηκαν
  - ▶ υπολογίζει την επόμενη κατάσταση της διεργασίας



## Έναρξη εκτέλεσης, Βήματα και Γύροι

- ▶ Αρχικά
  - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
  - ▶ όλα τα κανάλια είναι άδεια
- ▶ Όλες οι διεργασίες, επαναλαμβάνουν `συντονισμένα` τα ακόλουθα δύο βήματα:

### 1<sup>ο</sup> Βήμα

1. Εφαρμογή της γεννήτριας μηνυμάτων
2. Παραγωγή μηνυμάτων για τους εξερχόμενους γείτονες
3. Αποστολή μηνυμάτων μέσω των αντίστοιχων καναλιών

### 2<sup>ο</sup> Βήμα

1. Εφαρμογή της συνάρτησης αλλαγής κατάστασης
2. Διαγραφή όλων των μηνυμάτων από τα κανάλια.

- ▶ Ο συνδυασμός των δύο βημάτων ονομάζεται **γύρος**



## Μέτρηση πολυπλοκότητας

- ▶ Σχεδιασμός Συστήματος
  - ▶ Ορισμός Ελάχιστων Απαιτήσεων
  - ▶ Επιλογή κατάλληλου κατανεμημένου αλγόριθμου
  - ▶ Πως μπορούμε να μετρήσουμε την απόδοση;

### Χρονική πολυπλοκότητα

Το πλήθος των γύρων που απαιτούνται για να παραχθούν όλες οι ζητούμενες έξοδοι, ή μέχρι να τερματιστούν όλες οι διεργασίες (δηλ. να βρεθούν σε μια τερματική κατάσταση).

### Πολυπλοκότητα επικοινωνίας

Ο συνολικός αριθμός μη μηδενικών μηνυμάτων (δηλ. δεν προσμετρούνται τα `null` μηνύματα) που αποστέλλονται.



## Αναζήτηση κατά Εύρος

### Αναζήτηση κατά Εύρος

Σε ένα σύγχρονο δίκτυο  $G$ , η αναζήτηση κατά εύρος απαιτεί την κατασκευή ενός επικαλυπτικού δέντρου  $T(G)$ , με ρίζα την διεργασία  $u_0$  όπου οι κορυφές που είναι σε απόσταση  $d$  από την  $u_0$  στο  $G$ , βρίσκονται στο επίπεδο  $d$  στο δέντρο  $T(G)$ .

- ▶ Η κατασκευή αυτής της δομής είναι χρήσιμη σε πολλές περιπτώσεις
  - ▶ Γρήγορη μετάδοση πληροφορίας
  - ▶ Πρόβλημα Εκλογής Αρχηγού
  - ▶ Πρόβλημα Καταμέτρησης



## Ο Αλγόριθμος SynchBFS

### Αλγόριθμος SynchBFS

Οι διεργασίες διατηρούν μια μεταβλητή **μαρκαρισμένη** η οποία αρχικά είναι `false` και μια μεταβλητή **γονέας** με αρχική τιμή `0`. Αρχικά, η διεργασία  $u_0$  θέτει την μεταβλητή **μαρκαρισμένη** ως `true`, την μεταβλητή **γονέας** με την ταυτότητα της, και στέλνει ένα μήνυμα `αναζήτησης` σε όλους τους γείτονες της. Σε κάθε γύρο, εάν μια διεργασία λάβει ένα μήνυμα `αναζήτησης` και η τιμή της μεταβλητής **μαρκαρισμένη** είναι `false`, τότε θέτει την μεταβλητή σε `true`, θέτει την μεταβλητή **γονέας** με την ταυτότητα της διεργασίας από όπου έλαβε το μήνυμα, και στον επόμενο γύρο στέλνει ένα μήνυμα `αναζήτησης` σε όλους τους γείτονες της.

- ▶ Δεν γνωρίζουν το πλήθος των διεργασιών ( $n$ )
- ▶ Έχουν μοναδικές ταυτότητες



## Παράδειγμα Εκτέλεσης Αλγόριθμου ΣηψcBFS

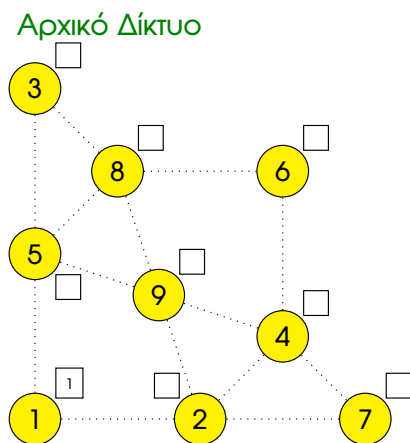
### Αρχικό Δίκτυο

Το δίκτυο έχει 9 μονάδες, 14 κανάλια

Η διεργασία 1 ξεκινά την εκτέλεση του αλγορίθμου.

Η διεργασία 1 θεωρείται μαρκαρισμένη.

Όλες οι άλλες διεργασίες δεν είναι μαρκαρισμένες.

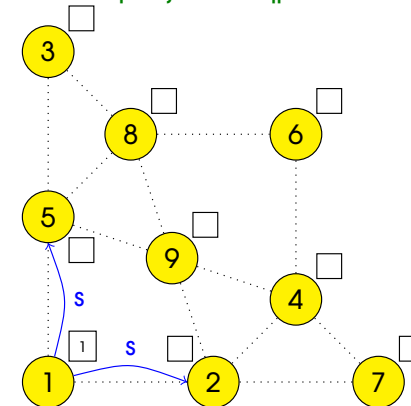


## Παράδειγμα Εκτέλεσης Αλγόριθμου ΣηψcBFS

### 1<sup>ος</sup> Γύρος – 1<sup>ο</sup> Βήμα

#### 1<sup>ος</sup> Γύρος – 1<sup>ο</sup> Βήμα

Η διεργασία 1 στέλνει μήνυμα 'αναζήτησης' σε όλους του γείτονες της.



## Παράδειγμα Εκτέλεσης Αλγόριθμου ΣηψcBFS

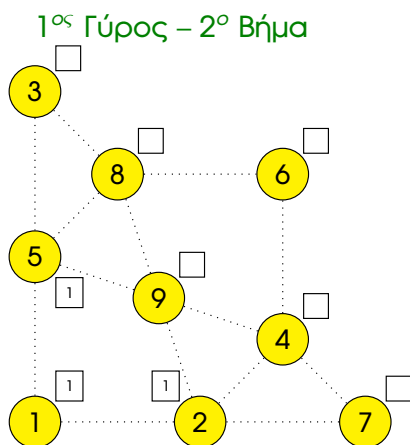
### 1<sup>ος</sup> Γύρος – 1<sup>ο</sup> Βήμα

Η διεργασία 1 στέλνει μήνυμα 'αναζήτησης' σε όλους του γείτονες της.

### 1<sup>ος</sup> Γύρος – 2<sup>ο</sup> Βήμα

Οι διεργασίες 2, 5 μαρκάρονται.

Οι διεργασίες 2, 5 θέτουν την 1 ως γονέα στο δέντρο.

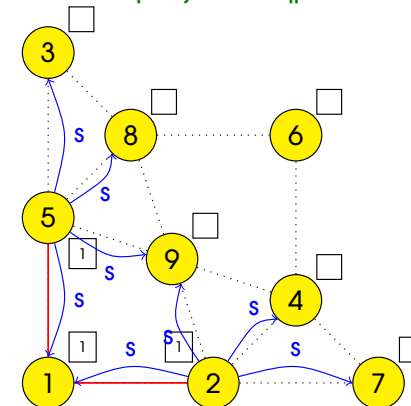


## Παράδειγμα Εκτέλεσης Αλγόριθμου ΣηψcBFS

### 2<sup>ος</sup> Γύρος – 1<sup>ο</sup> Βήμα

#### 2<sup>ος</sup> Γύρος – 1<sup>ο</sup> Βήμα

Οι διεργασίες 2, 5 στέλνουν μήνυμα 'αναζήτησης' σε όλους τους γείτονες τους.



## Παράδειγμα Εκτέλεσης Αλγόριθμου ΣyncBFS

### 2<sup>ος</sup> Γύρος – 1<sup>ο</sup> Βήμα

Οι διεργασίες **2, 5** στέλνουν μήνυμα 'αναζήτησης' σε όλους τους γείτονες τους.

### 2<sup>ος</sup> Γύρος – 2<sup>ο</sup> Βήμα

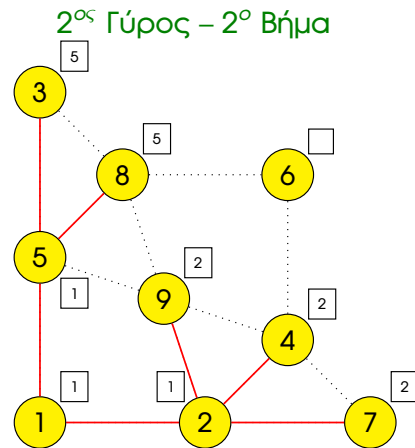
Η διεργασία **1** αγνοεί τα μηνύματα 'αναζήτησης' που έλαβε.

Οι διεργασίες **3, 4, 7, 8, 9** μαρκάρονται.

Οι διεργασίες **3, 8** θέτουν την **5** ως γονέα στο δέντρο.

Οι διεργασίες **4, 7** θέτουν την **2** ως γονέα στο δέντρο.

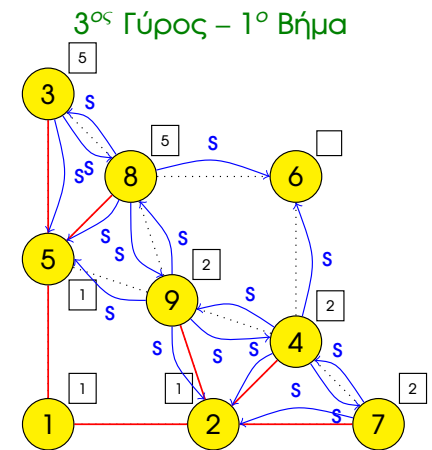
Η διεργασία **9** διαλέγει τυχαία την **2** ως γονέα στο δέντρο.



## Παράδειγμα Εκτέλεσης Αλγόριθμου ΣyncBFS

### 3<sup>ος</sup> Γύρος – 1<sup>ο</sup> Βήμα

Οι διεργασίες **3, 4, 7, 8, 9** στέλνουν μήνυμα 'αναζήτησης' σε όλους τους γείτονες τους.



## Παράδειγμα Εκτέλεσης Αλγόριθμου ΣyncBFS

### 3<sup>ος</sup> Γύρος – 1<sup>ο</sup> Βήμα

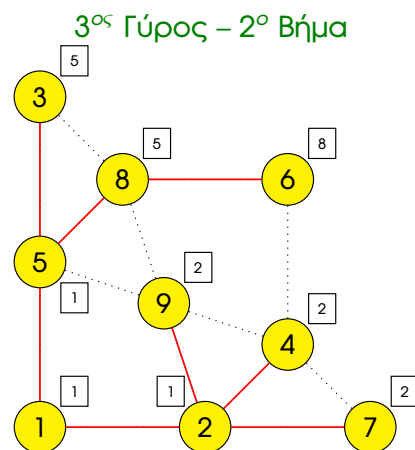
Οι διεργασίες **3, 4, 7, 8, 9** στέλνουν μήνυμα 'αναζήτησης' σε όλους τους γείτονες τους.

### 3<sup>ος</sup> Γύρος – 2<sup>ο</sup> Βήμα

Οι διεργασίες **2, 3, 4, 5, 7, 8, 9** αγνοούν τα μηνύματα 'αναζήτησης' που έλαβαν.

Η διεργασία **6** μαρκάρεται.

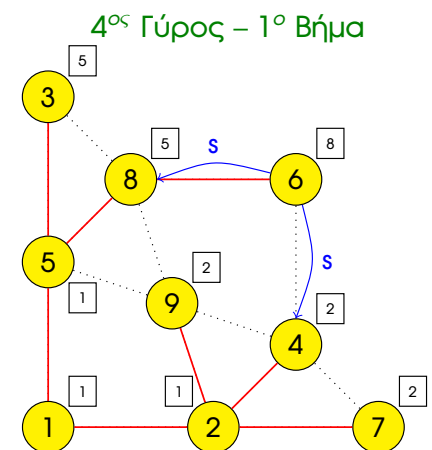
Η διεργασία **6** διαλέγει τυχαία την **8** ως γονέα στο δέντρο.



## Παράδειγμα Εκτέλεσης Αλγόριθμου ΣyncBFS

### 4<sup>ος</sup> Γύρος – 1<sup>ο</sup> Βήμα

Η διεργασία **6** στέλνει μήνυμα 'αναζήτησης' σε όλους του γείτονες της.





## Ο Αλγόριθμος BellmanFord

### Αλγόριθμος BellmanFord

Οι διεργασίες διατηρούν μια μεταβλητή **απόσταση** η οποία αρχικά είναι  $\infty$  και μια μεταβλητή **γονέας** με αρχική τιμή 0. Σε κάθε γύρο, οι διεργασίες στέλνουν την τιμή της μεταβλητής **απόσταση** σε όλους τους γείτονες τους. Κάθε διεργασία  $u$ , επεξεργάζεται τα μηνύματα που έλαβε, και εντοπίζει το μήνυμα της  $v$  το οποίο έχει την ελάχιστη τιμή **απόσταση<sub>v</sub> + βάρος<sub>vu</sub>**. Εάν ένα τέτοιο μήνυμα βρεθεί, θέτει την μεταβλητή **απόσταση<sub>u</sub>** στην τιμή **απόσταση<sub>v</sub> + βάρος<sub>vu</sub>** και την μεταβλητή **γονέας<sub>u</sub>** =  $v$ . Μετά από  $n - 1$  γύρους, η τιμή της μεταβλητής **απόσταση** είναι η ελάχιστη απόσταση της διεργασίας με την  $u_0$  και η μεταβλητή **γονέας** δείχνει τον γονέα της διεργασίας στο δέντρο των συντομότερων μονοπατιών.



## Σύνοψη 5<sup>ης</sup> Διάλεξης

### Προηγούμενο Μάθημα

Προηγούμενο Μάθημα

Σύγχρονα Κατανεμημένα Συστήματα

Κατανεμημένες Δομές

### Κατανεμημένες Δομές

Αναζήτηση κατά Βάθος

Ελάχιστα Επικαλυπτικά Δέντρα

### Σύνοψη Μαθήματος

Σύνοψη Μαθήματος

Βιβλιογραφία

Επόμενο Μάθημα



## Ο Αλγόριθμος DFS

### Αλγόριθμος SyncDFS

Οι διεργασίες διατηρούν μια μεταβλητή **μαρκαρισμένη** η οποία αρχικά είναι `false` και μια μεταβλητή **γονέας** με αρχική τιμή 0. Αρχικά, η διεργασία  $u_0$  θέτει την μεταβλητή **μαρκαρισμένη** ως `true`, την μεταβλητή **γονέας** με την ταυτότητα της, και στέλνει ένα μήνυμα "αναζήτησης" σε μία από τις γειτονικές της διεργασίες (έστω αυτή με τη μικρότερη ταυτότητα). Σε κάθε γύρο, εάν μια διεργασία λάβει ένα μήνυμα "αναζήτησης" και η τιμή της μεταβλητής **μαρκαρισμένη** είναι `false`, τότε θέτει την μεταβλητή σε `true`, θέτει την μεταβλητή **γονέας** με την ταυτότητα της διεργασίας από όπου έλαβε το μήνυμα, και στον επόμενο γύρο στέλνει ένα μήνυμα "αναζήτησης" σε μία από τις γειτονικές της διεργασίες.

- ▶ Δεν γνωρίζουν το πλήθος των διεργασιών ( $n$ )
- ▶ Έχουν μοναδικές ταυτότητες



## Ο Αλγόριθμος DFS

### Συνθήκες του Αλγορίθμου

Κάθε διεργασία, κάθε φορά που λαμβάνει το μήνυμα αναζήτησης:

1. Αν δεν έχει θέσει τη μεταβλητή γονέα, τότε θέτει ως γονέα τη διεργασία από την οποία έλαβε το μήνυμα.
2. Αν υπάρχουν γειτονικές διεργασίες (εκτός του γονέα) στις οποίες δεν έχει στείλει ακόμα το μήνυμα, τότε, επιλέγει μία εξ' αυτών και της στέλνει το μήνυμα. Αλλιώς, στέλνει το μήνυμα πίσω στον γονέα της.
3. Όταν μια διεργασία που έχει ήδη θέσει γονέα λάβει μήνυμα, τότε το στέλνει πίσω στην διεργασία από την οποία το έλαβε, αν αυτό επιτρέπεται από τις συνθήκες 1 και 2.

Συνθήκες 1 και 2 : αλγόριθμος **Tarry**

- ▶ Επικαλυπτικό δέντρο αλλά όχι κατα βάθος

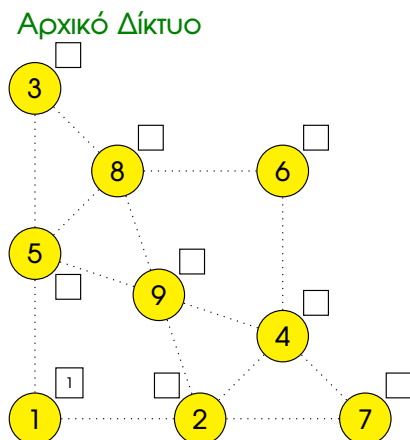


## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncDFS

### Αρχικό Δίκτυο

Το δίκτυο έχει 9 μονάδες, 14 κανάλια  
Η διεργασία 1 ξεκινά την εκτέλεση του αλγορίθμου.

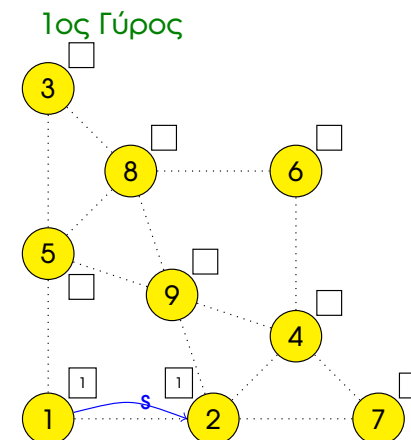
Η διεργασία 1 θεωρείται μαρκαρισμένη.  
Όλες οι άλλες διεργασίες δεν είναι μαρκαρισμένες.



## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncDFS

### 1ος Γύρος

Η διεργασία 1 στέλνει το μήνυμα στην γειτονική της διαδικασία με το μικρότερο id.

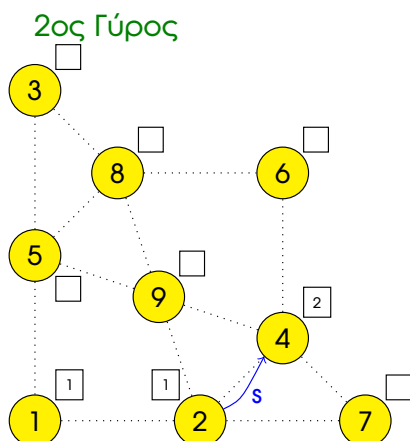


## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncDFS

### 2ος Γύρος

Η διεργασία 2 λαμβάνει το μήνυμα από την 1 και εφόσον δεν έχει γονέα, θέτει την 1 ως γονέα της.

Στη συνέχεια προωθεί το μήνυμα στην γειτονική της διαδικασία με το μικρότερο id - 4

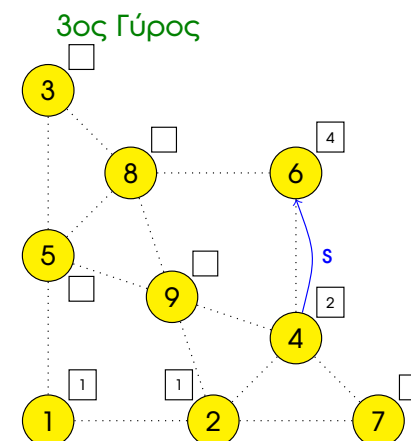


## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncDFS

### Επόμενοι Γύροι

Όποια διεργασία λαμβάνει μήνυμα από, θέτει τη διεργασία από την οποία το έλαβε ως γονέα και

στη συνέχεια προωθεί το μήνυμα στην γειτονική της διαδικασία με το μικρότερο id

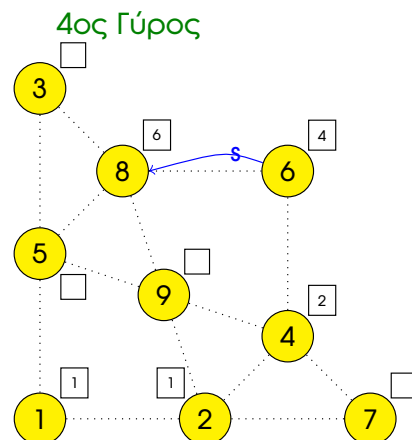


## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncDFS

### Επόμενοι Γύροι

Όποια διεργασία λαμβάνει μήνυμα από, θέτει τη διεργασία από την οποία το έλαβε ως γονέα και

στη συνέχεια προωθεί το μήνυμα στην γειτονική της διαδικασία με το μικρότερο id

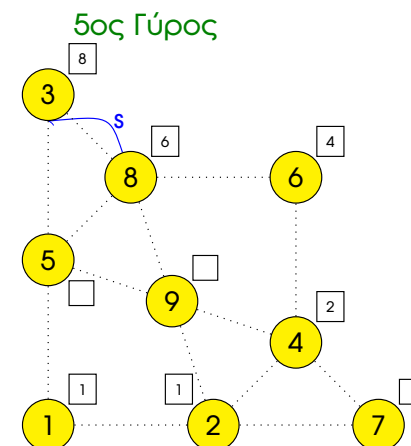


## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncDFS

### Επόμενοι Γύροι

Όποια διεργασία λαμβάνει μήνυμα από, θέτει τη διεργασία από την οποία το έλαβε ως γονέα και

στη συνέχεια προωθεί το μήνυμα στην γειτονική της διαδικασία με το μικρότερο id

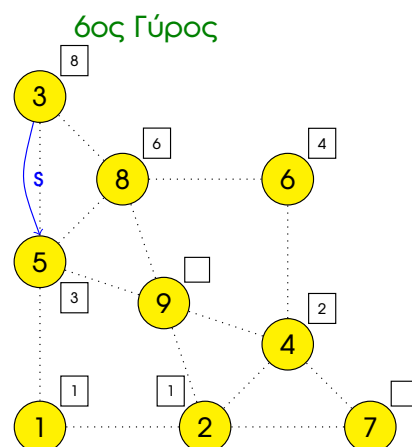


## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncDFS

### Επόμενοι Γύροι

Όποια διεργασία λαμβάνει μήνυμα από, θέτει τη διεργασία από την οποία το έλαβε ως γονέα και

στη συνέχεια προωθεί το μήνυμα στην γειτονική της διαδικασία με το μικρότερο id

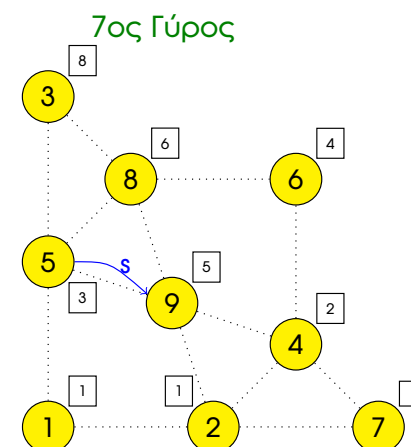


## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncDFS

### Επόμενοι Γύροι

Όποια διεργασία λαμβάνει μήνυμα από, θέτει τη διεργασία από την οποία το έλαβε ως γονέα και

στη συνέχεια προωθεί το μήνυμα στην γειτονική της διαδικασία με το μικρότερο id

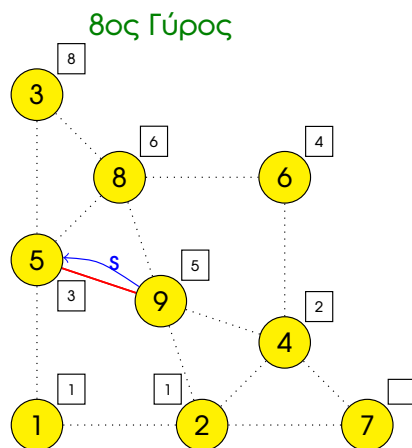


## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncDFS

### 8ος Γύρος

Η διαδικασία 9 δεν διαθέτει γείτονες οι οποίοι δεν έχει λάβει το μήνυμα.

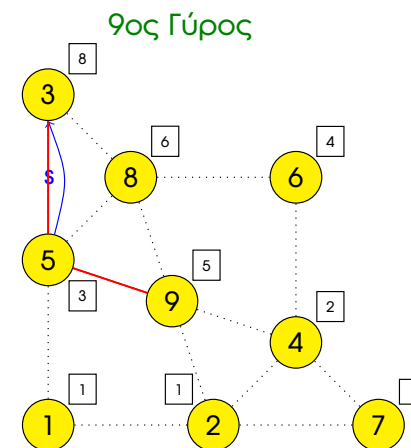
Στέλνει το μήνυμα πίσω στον γονέα της.



## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncDFS

### Επόμενοι Γύροι

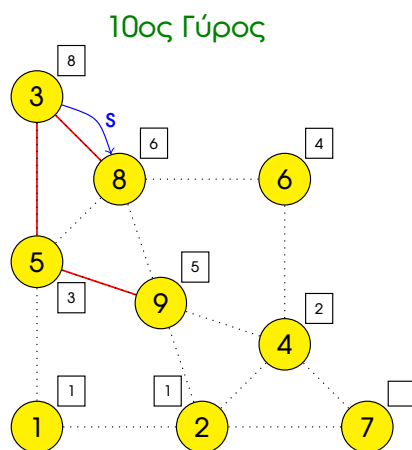
Κάθε διαδικασία που λαμβάνει το μήνυμα χωρίς εξερχόμενες ακμές που δεν έχει χρησιμοποιήσει, προωθεί το μήνυμα πίσω στον γονέα της



## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncDFS

### Επόμενοι Γύροι

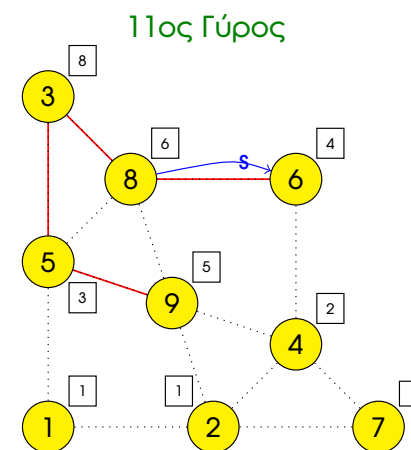
Κάθε διαδικασία που λαμβάνει το μήνυμα χωρίς εξερχόμενες ακμές που δεν έχει χρησιμοποιήσει, προωθεί το μήνυμα πίσω στον γονέα της



## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncDFS

### Επόμενοι Γύροι

Κάθε διαδικασία που λαμβάνει το μήνυμα χωρίς εξερχόμενες ακμές που δεν έχει χρησιμοποιήσει, προωθεί το μήνυμα πίσω στον γονέα της

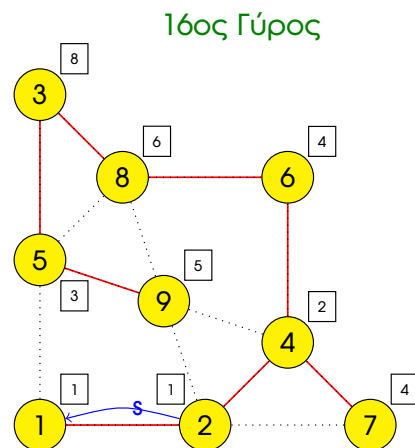




## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncDFS

### 16ος Γύρος

Η διαδικασία 2 προωθεί το μήνυμα στη ρίζα του δέντρου που δημιουργήθηκε ο ο αλγόριθμος τερματίζει έχοντας πραγματοποιήσει μία αναζήτηση κατά βάθος.



## Χαρακτηριστικά του Αλγόριθμου SyncDFS

- ▶ Ο αλγόριθμος SyncDFS κατασκευάζει ένα επικαλυπτικό δέντρο αναζήτησης κατά βάθος.
  - ▶ Αποφεύγονται πιθανοί κύκλοι λόγω της Συνθήκης 3
- ▶ Η δομή του δέντρου δεν είναι αποθηκευμένη σε κάποια 'κεντρική' διεργασία
- ▶ Η χρονική πολυπλοκότητα είναι  $\mathcal{O}(2|E|)$ 
  - ▶ Κάθε ακμή του δικτύου χρησιμοποιείται το πολύ δύο φορές αφού καμιά διεργασία δεν στέλνει το μήνυμα στον ίδιο γείτονα δύο φορές. (Συνθήκη 1)
- ▶ Η πολυπλοκότητα επικοινωνίας είναι επίσης  $\mathcal{O}(2|E|)$ 
  - ▶ Κάθε φορά μόνο μία διεργασία 'έχει' το μήνυμα και μπορεί να το στείλει
  - ▶ Τα μηνύματα ανταλλάσσονται σειριακά



## Ο Αλγόριθμος GHS

- ▶ Πήρε το όνομά του από τους τρεις δημιουργούς του : **G**allager, **H**umbert, **S**pira
- ▶ Αρχικός σχεδιασμός για ασύγχρονα δίκτυα
- ▶ Δημιουργία ενός πλήρως συνδεδεμένου γράφου
- ▶ Εκτελείται σε φάσεις/επίπεδα
  - ▶ Δημιουργούνται διακριτά fragments
  - ▶ Σταδιακά συγχωνεύονται και τελικά καλύπτεται όλο το δίκτυο των διεργασιών



## Ο Αλγόριθμος GHS

### Αλγόριθμος GHS

Αρχικά όλες οι διεργασίες βρίσκονται στην κατάσταση **Sleeping**. Μια τυχαία διεργασία  $u_0$  ξεκινά την εκτέλεση στέλνοντας μήνυμα **αρχικοποίησης** στις γειτονικές της διεργασίες. Όταν μία διεργασία λάβει μήνυμα **αρχικοποίησης** ενεργοποιείται, μεταβαίνει στην κατάσταση **Find** και συμμετέχει στην αναζήτηση του **MST** στέλνοντας μηνύματα αρχικοποίησης στις γειτονικές της διεργασίες. Η τελευταία κατάσταση στην οποία μπορεί να μεταβεί μια διεργασία είναι η **Found** στην οποία εισέρχεται μια διεργασία όταν δεν διαθέτει πλέον εξερχόμενες ακμές ή έχει βρεθεί η εξερχόμενη ακμή με το ελάχιστο βάρος για εκείνη.

- ▶ Είναι γνωστός ο αριθμός των διεργασιών του δικτύου ( $n$ )
- ▶ Οι διεργασίες έχουν μοναδικές ταυτότητες
- ▶ Είναι γνωστό το βάρος των ακμών που προσπίπτουν σε κάθε διεργασία
- ▶ Συγκριση μεταξύ βαρών για την επιλογή του μικρότερου
- ▶ Ο αριθμός των επιπέδων είναι προκαθορισμένος και γνωστός



## Κατάσταση διεργασιών

- ▶ Sleeping
- ▶ Find
- ▶ Found

## Καταστάσεις ακμών

- ▶ Branch
- ▶ Rejected
- ▶ Basic

## Τύποι μηνύματων

- ▶ Connect
- ▶ Initiate
- ▶ Test
- ▶ Accept
- ▶ Reject
- ▶ Report
- ▶ Change-core

## Αρχικό Δίκτυο

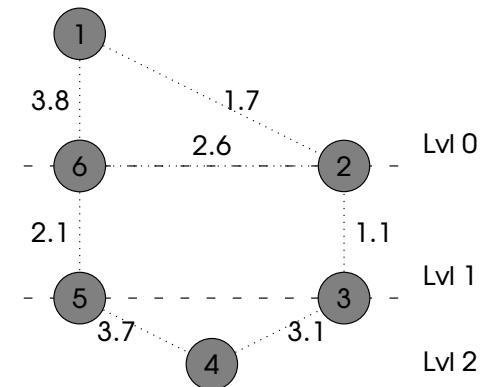
Το δίκτυο έχει 6 διεργασίες, 7 μη κατευθυνόμενα κανάλια

Οι διεργασίες έχουν μοναδικές ταυτότητες

Ταξινομούνται σε επίπεδα Level 0,1,2

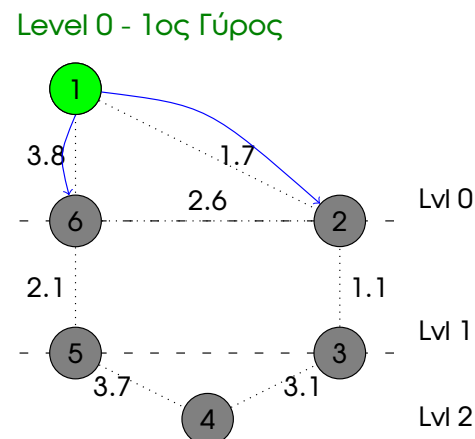
Κάθε ακμή έχει διακριτό βάρος και είναι γνωστό στους κόμβους που προσπίπει

## Αρχικό Δίκτυο



## Level 0

Η διεργασία 1 στέλνει μηνύματα initiate στους γειτονικούς κόμβους.

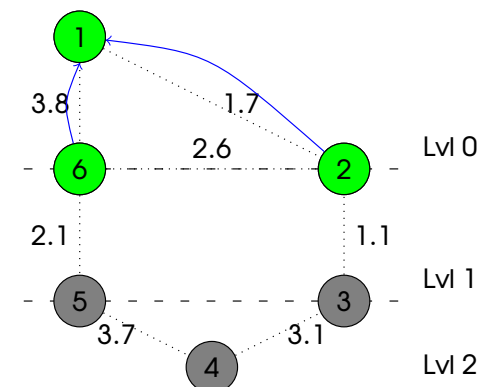


## Level 0

Η διεργασία 1 στέλνει μηνύματα initiate στους γειτονικούς κόμβους.

Οι διεργασίες 2, 6 ενεργοποιούνται και απαντάνε με report στην 1.

## Level 0 - 2ος Γύρος



## Παράδειγμα Εκτέλεσης Αλγόριθμου SynchGHS

### Level 0

Η διεργασία 1 στέλνει μηνύματα *initiate* στους γειτονικούς κόμβους.

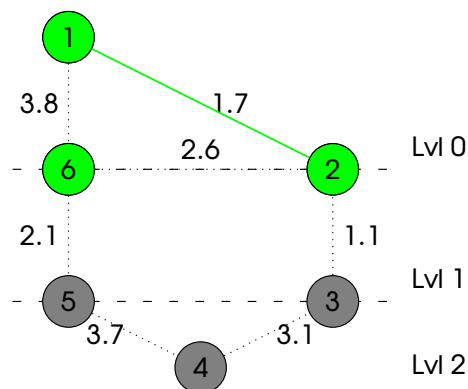
Οι διεργασίες 2, 6 ενεργοποιούνται και απαντάνε με *report* στην 1.

Η 1 στέλνει *connect* στη 2 καθώς η ακμή (1, 2) έχει το μικρότερο βάρος.

Η 2 έχει πλέον την 1 ως γονέα.

Εμφανίζονται δύο *fragments*.

### Level 0 - 3ος Γύρος

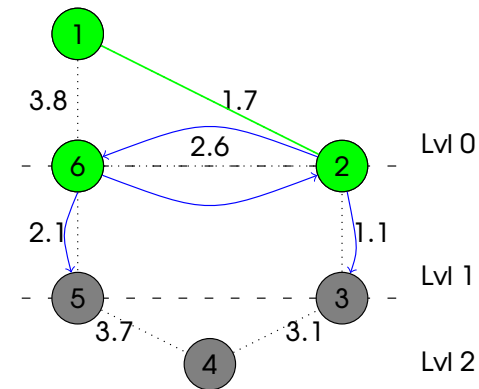


## Παράδειγμα Εκτέλεσης Αλγόριθμου SynchGHS

### Level 1 - 1ος Γύρος

### Level 1

Οι 2, 6 στέλνουν *initiate* στις γειτονικές τους διεργασίες.



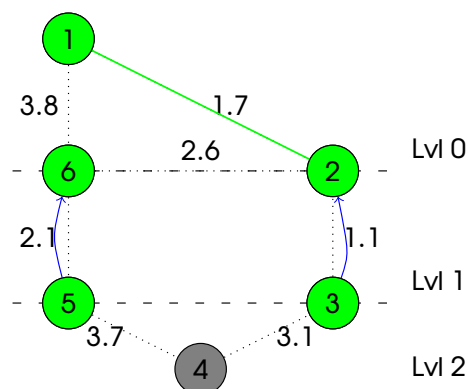
## Παράδειγμα Εκτέλεσης Αλγόριθμου SynchGHS

### Level 1

Οι 2, 6 στέλνουν *initiate* στις γειτονικές τους διεργασίες.

Οι 5, 3 ενεργοποιούνται και απαντάνε με *report* στους 6, 2 αντίστοιχα.

### Level 1 - 2ος Γύρος



## Παράδειγμα Εκτέλεσης Αλγόριθμου SynchGHS

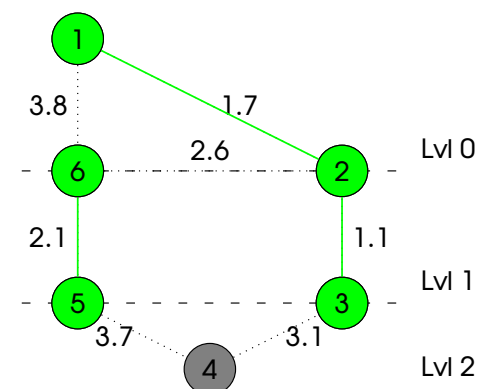
### Level 1

Οι 2, 6 στέλνουν *initiate* στις γειτονικές τους διεργασίες.

Οι 5, 3 ενεργοποιούνται και απαντάνε με *report* στους 6, 2 αντίστοιχα.

Οι 6 στέλνει *connect* στον 5 και ο 2 στον 3, αντίστοιχα.

### Level 1 - 3ος Γύρος



## Παράδειγμα Εκτέλεσης Αλγόριθμου SynchGHS

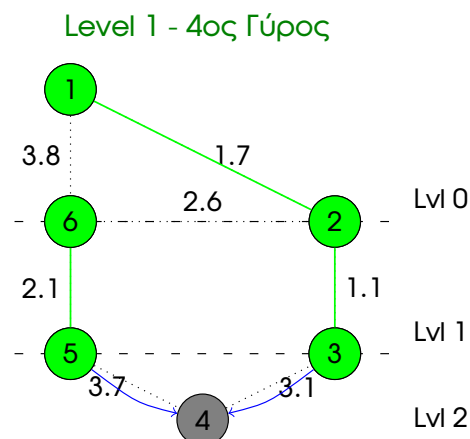
### Level 1

Οι 2, 6 στέλνουν *initiate* στις γειτονικές τους διεργασίες.

Οι 5, 3 ενεργοποιούνται και απαντάνε με *report* στους 6, 2 αντίστοιχα.

Οι 6 στέλνει *connect* στον 5 και ο 2 στον 3, αντίστοιχα.

Οι 5, 3 στέλνουν *init* στον 4.



## Παράδειγμα Εκτέλεσης Αλγόριθμου SynchGHS

### Level 1

Οι 2, 6 στέλνουν *initiate* στις γειτονικές τους διεργασίες.

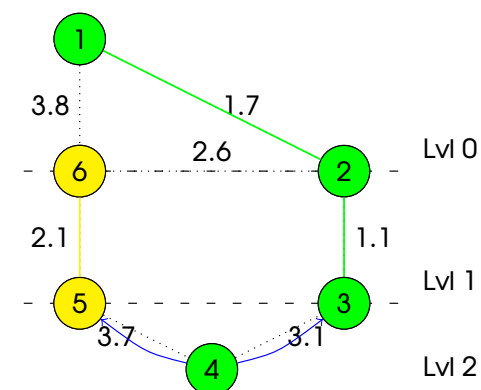
Οι 5, 3 ενεργοποιούνται και απαντάνε με *report* στους 6, 2 αντίστοιχα.

Οι 6 στέλνει *connect* στον 5 και ο 2 στον 3, αντίστοιχα.

Οι 5, 3 στέλνουν *init* στον 4.

Οι 4 απαντάει *report* συμπεριλαμβάνοντας τα στοιχεία του.

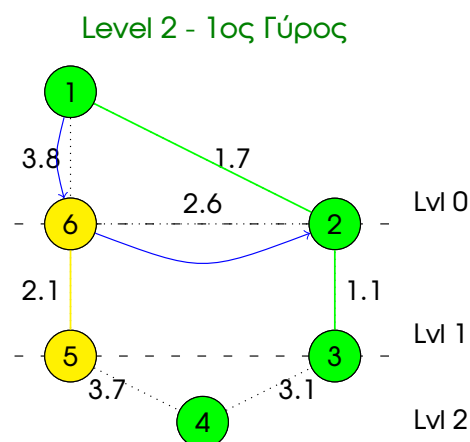
### Level 1 - 5ος Γύρος



## Παράδειγμα Εκτέλεσης Αλγόριθμου SynchGHS

### Level 2

Οι 1, 6 ως ρίζες *fragments* ξεκινούν τη διαδικασία συνένωσης στέλνοντας *test* στις γειτονικές διεργασίες που δεν ανήκουν στο ίδιο *fragment*.



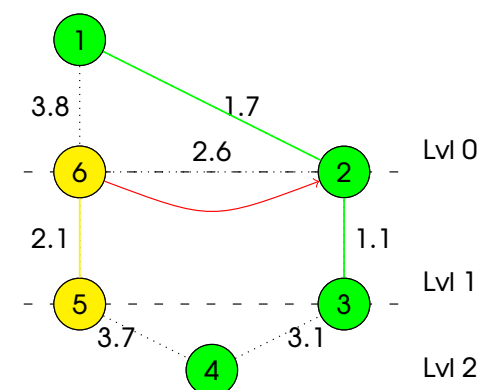
## Παράδειγμα Εκτέλεσης Αλγόριθμου SynchGHS

### Level 2

Οι 1, 6 ως ρίζες *fragments* ξεκινούν τη διαδικασία συνένωσης στέλνοντας *test* στις γειτονικές διεργασίες που δεν ανήκουν στο ίδιο *fragment*.

Η 6 στέλνει *connect* στη 2 καθώς η ακμή (6, 2) έχει το μικρότερο βάρος.

### Level 2 - 2ος Γύρος



## Παράδειγμα Εκτέλεσης Αλγόριθμου SynchGHS

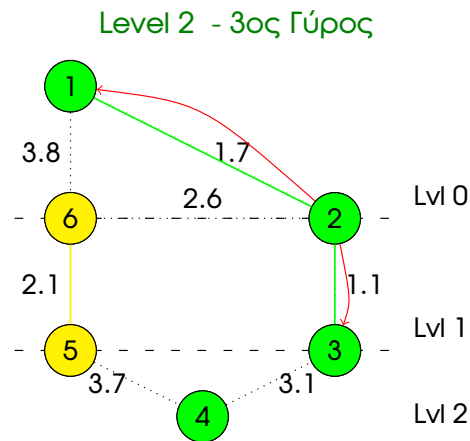
### Level 2

Οι 1, 6 ως ρίζες fragments ξεκινούν τη διαδικασία συνένωσης στέλλοντας test στις γειτονικές διεργασίες που δεν ανήκουν στο ίδιο fragment.

Η 6 στέλνει connect στη 2 καθώς η ακμή (6, 2) έχει το μικρότερο βάρος.

Πριν γίνει η συνένωση, η 2 πρέπει να γίνει ρίζα του fragment στο οποίο ανήκει.

Έτσι, η 2 στέλνει change core στις γειτονικές του ίδιου fragment.

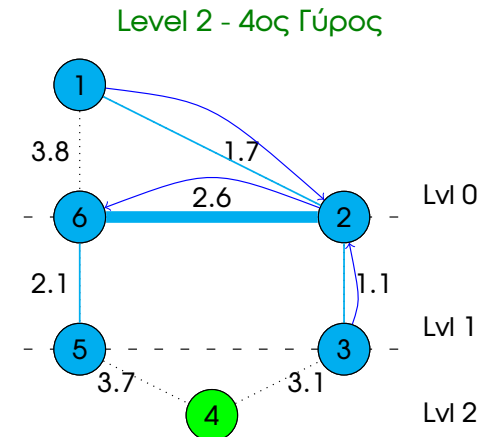


## Παράδειγμα Εκτέλεσης Αλγόριθμου SynchGHS

### Level 2

Οι 1, 3 απαντούν accept ενημερώνοντας τη 2 ότι την έχουν θέσει ως γονέα.

Η 2 απαντάει στον 6 accept για το connect του προηγούμενου γύρου.



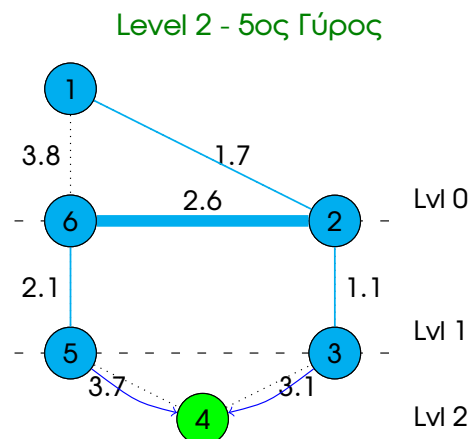
## Παράδειγμα Εκτέλεσης Αλγόριθμου SynchGHS

### Level 2

Οι 1, 3 απαντούν accept ενημερώνοντας τη 2 ότι την έχουν θέσει ως γονέα.

Η 2 απαντάει στον 6 accept για το connect του προηγούμενου γύρου.

Οι 3, 5 στέλνουν connect στην 4.



## Παράδειγμα Εκτέλεσης Αλγόριθμου SynchGHS

### Level 2

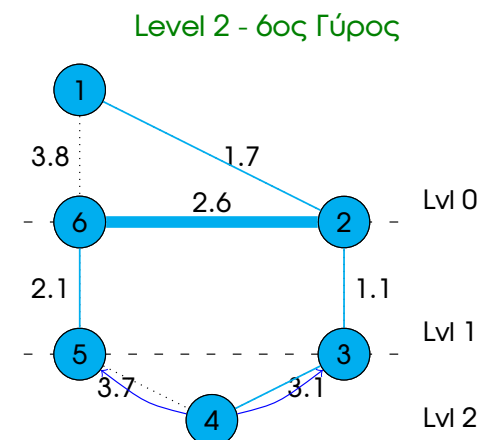
Οι 1, 3 απαντούν accept ενημερώνοντας τη 2 ότι την έχουν θέσει ως γονέα.

Η 2 απαντάει στον 6 accept για το connect του προηγούμενου γύρου.

Οι 3, 5 στέλνουν connect στην 4.

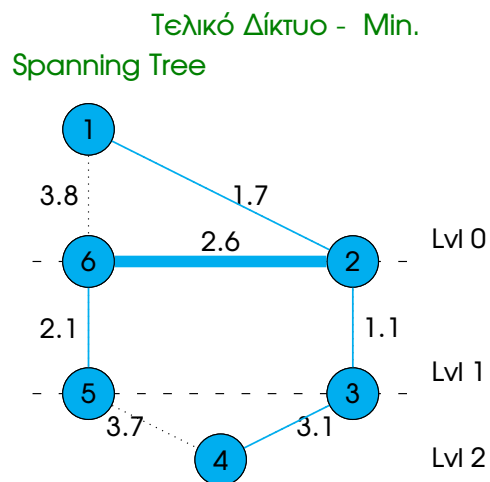
Η 4 επιλέγει την 3 ως γονέα λόγω βάρους ακμής.

Η 4 απαντάει accept στην 3 και reject στην 5.



## Παράδειγμα Εκτέλεσης Αλγόριθμου SynchGHS

Ο αλγόριθμος τερματίζει καθώς η 4 δεν έχει άλλες εξερχόμενες ακμές.  
Έχει σημασιστεί το ζητούμενο Minimum Spanning Tree.



## Χαρακτηριστικά του Αλγόριθμου SynchGHS

- ▶ Ο αλγόριθμος SynchGHS
- ▶ Η δομή του δέντρου δεν είναι αποθηκευμένη σε κάποια "κεντρική" διεργασία
- ▶ Η χρονική πολυπλοκότητα είναι  $\mathcal{O}(n \log n)$
- ▶ Η πολυπλοκότητα επικοινωνίας είναι  $\mathcal{O}((n + |E|) \log n)$ 
  - ▶ Η μέγιστη τιμή που μπορεί να πάρει είναι  $5n \log n + 2|E|$
  - ▶  $2E$ ,  $E$  Ο μέγιστος και ο ελάχιστος αριθμός δοκιμαστικών μηνυμάτων που μπορούν να αποσταλούν



## Κατασκευή Minimum Spanning Tree

Δοσμένου ενός fragment ενός MST, έστω  $e$  η εξερχόμενη ακμή με το μικρότερο βάρος του fragment. Συγχωνεύοντας την διεργασία στην οποία προσπίπτει η  $e$  και δεν ανήκει στο fragment, δημιουργείται ένα άλλο fragment του MST.

- ▶ Έστω η προστιθέμενη ακμή  $e$  δεν περιέχεται στο αρχικό MST
- ▶ Υπάρχει κύκλος που περιλαμβάνει την  $e$  και ένα υποσύνολο ακμών του MST.
- ▶ Τουλάχιστον μια ακμή  $x \neq e$  του κύκλου αυτού είναι επίσης εξερχόμενη ακμή του fragment.
- ▶ Έτσι θα ισχύει  $w(x) \geq w(e)$ .
- ▶ Διαγράφοντας την ακμή  $x$  από το MST και προσθέτοντας την ακμή  $e$  δημιουργείται ένα νέο spanning tree το οποίο πρέπει να είναι ελάχιστο εφόσον το αρχικό δένδρο ήταν ελάχιστο.
- ▶ Το αρχικό fragment με την προσθήκη της  $e$  είναι το fragment του νέου MST.



## Σύνοψη 5ης Διάλεξης

### Προηγούμενο Μάθημα

Προηγούμενο Μάθημα  
Σύγχρονα Κατανεμημένα Συστήματα  
Κατανεμημένες Δομές

### Κατανεμημένες Δομές

Αναζήτηση κατά Βάθος  
Ελάχιστα Επικαλυπτικά Δέντρα

### Σύνοψη Μαθήματος

Σύνοψη Μαθήματος  
Βιβλιογραφία  
Επόμενο Μάθημα



## Σύνοψη Μαθήματος

- ▶ Σύγχρονα Κατανεμημένα Συστήματα
- ▶ Αναζήτηση κατά Βάθος
  - ▶ Αλγόριθμος SyncDFS
- ▶ Ελάχιστα Επικαλυπτικά Δέντρα
  - ▶ Αλγόριθμος των **G**allager, **H**umbert, **S**pira



## Βιβλιογραφία

- ▶ Σημειώσεις του μαθήματος
- ▶ Βιβλίο "Distributed Algorithms" (N.Lynch)
  1. Κεφάλαιο 4: Algorithms in General Synchronous Networks
- ▶ Βιβλίο "Distributed Computing Fundamentals, Simulations, and Advanced Topics" (H.Attiya, J.Welch)
  1. Κεφάλαιο 2: Basic Algorithms in Message Passing Systems
- ▶ Βιβλίο "Introduction to Distributed Algorithms" (G.Tel)
  1. Κεφάλαιο 6: Wave and Traversal Algorithms
- ▶ Βιβλίο "Distributed Systems, Concepts and Design" (G.Coulouris, J.Dollimore, T.Kindberg)
  1. Κεφάλαιο 11: Coordination and Agreement



## Επόμενο Μάθημα

- ▶ Σύγχρονα Κατανεμημένα Συστήματα
- ▶ Συναίνεση Με Σφάλματα

