

## Κατανομημένα Συστήματα I

### Μάθημα Βασικής Επιλογής, Χειμερινού Εξαμήνου

#### Τομέας Εφαρμογών και Θεμελιώσεων

Ιωάννης Χατζηγιαννάκης

Παρασκευή, 23 Οκτωβρίου, 2009  
Αίθουσα Β3



## Αρχικές Υποθέσεις

- ▶ Οι ερωτήσεις αντιστοιχούν σε προβλήματα των ΚΣ
- ▶ Προσπαθούμε να περιγράψουμε το ΚΣ (όπου εμφανίζεται το πρόβλημα)
- ▶ Η περιγραφή γίνεται με αφαιρετικό τρόπο – οι αρχικές υποθέσεις
- ▶ Ένα πρόβλημα μπορεί να διαφοροποιηθεί αν αλλάξουν οι αρχικές υποθέσεις
  - ▶ είτε να γίνει ποιο εύκολο, είτε ποιο δύσκολο
  - ▶ ακόμα και τελείως διαφορετικό
- ▶ Προτού αρχίσουμε να εξετάζουμε το πρόβλημα πρέπει να καταγράψουμε όλες τις αρχικές υποθέσεις που γίνονται
- ▶ Πρέπει να κατανοήσουμε ακριβώς γιατί δίνεται μια αρχική υπόθεση



## Μοντέλο Επικοινωνίας

- ▶ **Σύγχρονο** δίκτυο επικοινωνίας ή **Ασύγχρονο** δίκτυο επικοινωνίας ;
  - ▶ Η πρώτη άσκηση αφορά το σύγχρονο μοντέλο
  - ▶ Η δεύτερη άσκηση αφορά το ασύγχρονο μοντέλο
- ▶ Η πρώτη διαφορά είναι το θέμα 'αυτονομίας' στην εκτέλεση των βημάτων
- ▶ Η ουσιαστική διαφορά είναι η χρονική απροσδιοριστία
  - ▶ ως προς τους χρόνους εκτέλεσης των βημάτων
  - ▶ ως προς τους χρόνους ανταλλαγής μηνυμάτων
- ▶ Για την μελέτη της απόδοσης στα ασύγχρονα δίκτυα πρέπει να υποθέσουμε κάποια άνω όρια
  - ▶ Η πολυπλοκότητα (χρονική ή μηνυμάτων) εκφράζεται πάντα με βάση αυτών των ορίων



## Τοπολογία Δικτύου

- ▶ Το δίκτυο μοντελοποιείται σύμφωνα με ένα γράφημα επικοινωνίας
  - ▶ Κορυφές αντιστοιχούν στις διεργασίες
  - ▶ Ακμές αντιστοιχούν στα κανάλια επικοινωνίας
- ▶ Ποιες είναι οι υποθέσεις για το γράφημα επικοινωνίας ;
- ▶ Είναι ιδιαίτερης κατηγορίας (π.χ. δακτύλιος) ή είναι γενικό ;
- ▶ Κατευθυνόμενο ή μη-κατευθυνόμενο ;
- ▶ Είναι πλήρως συνδεδεμένο ;
- ▶ Γνωρίζουμε το πλήθος των κορυφών και των ακμών ;
- ▶ Πρέπει να κατανοήσουμε ακριβώς το δίκτυο – η λύση μας μπορεί να είναι εντελώς λάθος
- ▶ Συνήθως τα προβλήματα υποθέτουν ένα γενικό, μη-κατευθυνόμενο δίκτυο επικοινωνίας με n διεργασίες και m κανάλια επικοινωνίας



## Μοντέλο Σφαλμάτων

- ▶ Υπάρχουν σφάλματα στο σύστημα ;
  - ▶ Σφάλματα στις διεργασίες
  - ▶ Σφάλματα στα κανάλια επικοινωνίας
- ▶ Τα σφάλματα είναι παροδικά ;
- ▶ Γνωρίζουμε το πλήθος των σφαλμάτων που μπορούν να συμβούν κατά την εκτέλεση του αλγορίθμου ;
- ▶ Πρέπει να κατανοήσουμε ακριβώς το μοντέλο σφαλμάτων – η λύση μας μπορεί να είναι εντελώς λάθος
- ▶ Η μελέτη της απόδοσης ίσως να πρέπει να γίνει σε σχέση με το πλήθος σφαλμάτων
- ▶ Συνήθως τα προβλήματα δεν κάνουν υποθέσεις σφαλμάτων – τα μελετήσαμε στην 4η διάλεξη



## Αρχικές Γνώσεις Διεργασιών

- ▶ Τι γνωρίζουν οι διεργασίες για το σύστημα ;
  - ▶ Την τοπολογία
  - ▶ Την διάμετρο
  - ▶ Το πλήθος των διεργασιών
- ▶ Οι διεργασίες έχουν (ή δεν έχουν) μοναδικές ταυτότητες
- ▶ Υπάρχει μια διεργασία που την γνωρίζουν όλες οι άλλες (π.χ.,  $u_0$ )
- ▶ Δίνεται είσοδος κάποια τιμή
  - ▶ ακέραιος αριθμός  $i_0$
  - ▶ σύμφωνα κάποιο σύνολο  $S$
- ▶ Πρέπει να κατανοήσουμε ακριβώς γιατί δίνεται (ή δεν δίνεται) ένα κομμάτι γνώσης



## Ζητούμενο – Πρόβλημα

- ▶ Συνήθως είναι προβλήματα σχεδιασμού ενός νέου αλγορίθμου
  - ▶ Υπάρχουν προβλήματα εντοπισμού καλύτερων / χειρότερων διατάξεων διεργασιών
  - ▶ Υπάρχουν προβλήματα απόδειξης συγκεκριμένων συνθηκών
- ▶ Ποιο είναι το ζητούμενο ;
  - ▶ Ποιο πρόβλημα θέλουμε να λύσουμε ;
  - ▶ Τι θέλουμε να 'μάθουν' οι διεργασίες ;
- ▶ Ταιριάζει σε κάποιο από τα προβλήματα που έχουμε μελετήσει ;
  - ▶ Πως διαφοροποιούνται οι αρχικές υποθέσεις ;
  - ▶ Μήπως στις σημειώσεις υποθέτουμε περισσότερη αρχική γνώση ;
  - ▶ Απαιτείται ιδιαίτερη τοπολογία που στην συγκεκριμένη ερώτηση δεν δίνεται ;
- ▶ Πρέπει ο αλγόριθμος να τερματίζει ;



## Καταγραφή Υποθέσεων / Μοντέλου

- ▶ Εντοπισμός αρχικών υποθέσεων
- ▶ Μοντέλο Επικοινωνίας
- ▶ Τοπολογία Δικτύου
- ▶ Σφάλματα
- ▶ Γνώσεις διεργασιών
- ▶ Κατανόηση Προβλήματος
- ▶ Χαρακτηρισμός Προβλήματος



## Αρχική Προσέγγιση

- ▶ Έχουμε κάποια διαίσθηση για την λύση ;
- ▶ Έχουμε πάρει κάποια πρώτη απόφαση για την λύση που θα δώσουμε ;
  - ▶ σκεφτείτε το ξανά !
  - ▶ μήπως μας έχει ξεφύγει κάτι ;
- ▶ Σκιαγραφήστε την λύση
  - ▶ ελέγχουμε αν συμφωνεί με τις αρχικές υποθέσεις
  - ▶ έχουμε χρησιμοποιήσει όλες τις πληροφορίες;
- ▶ Είναι η λύση σωστή ; μπορούμε να επικειρηματολογήσουμε ;
- ▶ Ποια είναι η πολυπλοκότητα (χρονική, επικοινωνίας) ;
- ▶ Ποια είναι η ανεκπικότητα σε σφάλματα ;
- ▶ Μήπως υπάρχει καλύτερη λύση ;



## Σχεδιασμός Λύσης

- ▶ Χρειάζεται να τρέξουμε κάποιο αλγόριθμο για να αποκτήσουμε κάποια πληροφορία ;
  - ▶ εκλογή αρχηγού για να προκύψει μια μοναδική διεργασία (π.χ.,  $u_2$ )
  - ▶ καταμέτρηση για να βρεθεί το πλήθος των διεργασιών ( $n$ )
  - ▶ υπολογισμός διαμέτρου
- ▶ Αν το δίκτυο είναι γενικό – απαιτείτε κάποια συγκεκριμένη τοπολογία ;
  - ▶ χρειάζομαστε ένα δένδρο / δακτύλιο ;
  - ▶ συνήθως κάνει το πρόβλημα 'πιο εύκολο' – ή την λύση 'πιο αποδοτική'
- ▶ Αποτύπωση γενικής ιδέας – 1 παράγραφος
  - ▶ ποια είναι η βασική ιδέα ;
  - ▶ τι κάνει ο αλγόριθμος ;
  - ▶ γιατί είναι σωστός ;



## Αποτύπωση Λύσης

- ▶ Αναλυτική καταγραφή μεταβλητών
  - ▶ σκοπός – χρήση
  - ▶ τύπος μεταβλητής
  - ▶ αρχική τιμή
- ▶ Αναλυτική καταγραφή μηνυμάτων
  - ▶ σκοπός – χρήση
  - ▶ περιεχόμενα μηνυμάτων
- ▶ Αρχικοποίηση συστήματος
  - ▶ δημιουργία κάποιας συγκεκριμένης 'εικονικής' τοπολογίας
  - ▶ εκτέλεση αλγόριθμου (απόκτηση γνώσης)
  - ▶ αρχικοποίηση μεταβλητών
- ▶ Βασικός γύρος εκτέλεσης
- ▶ Ειδικές περιπτώσεις
- ▶ Τερματισμός
- ▶ Άλλες πληροφορίες



## Τελική Απάντηση

1. Σύνομη Περιγραφή
2. Περιγραφή Διεργασιών
  - ▶ μεταβλητές
  - ▶ τύποι μηνυμάτων
  - ▶ αρχικοποίηση
3. Βοηθητικοί Αλγόριθμοι
4. Βασικός Αλγόριθμος – περιγραφή εκτέλεσης
  - ▶ 'απλός' / 'τυπικός' γύρος εκτέλεσης
  - ▶ ειδικές περιπτώσεις
5. Ψευδοκώδικα (αν υπάρχει χρόνος)
6. Ορθότητα – Συζήτηση ... Απόδειξη
7. Χρονική Πολυπλοκότητα – Συζήτηση ... Απόδειξη
8. Πολυπλοκότητα Μηνυμάτων – Συζήτηση ... Απόδειξη



## 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 Εικόνης Αρχηγού
  - ▶ Πρόβλημα Καταμέτρησης



## 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.

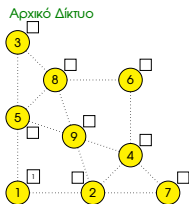
- ▶ Δεν γνωρίζουν το πλήθος των διεργασιών ( $n$ )
- ▶ Έχουν μοναδικές ταυτότητες



## Παράδειγμα Εκτέλεσης Αλγορίθμου SynchBFS

## Αρχικό Δίκτυο

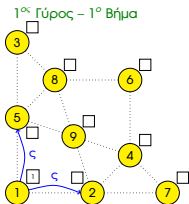
Το δίκτυο έχει 9 μονάδες, 14 κανάλια  
 Η διεργασία 1 ξεκινά την εκτέλεση του αλγορίθμου.  
 Η διεργασία 1 θεωρείται μαρκαρισμένη.  
 Όλες οι άλλες διεργασίες δεν είναι μαρκαρισμένες.



## Παράδειγμα Εκτέλεσης Αλγορίθμου SynchBFS

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

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



## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncBFS

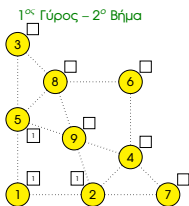
### 1<sup>ος</sup> Γύρος – 1<sup>ο</sup> Βήμα

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.

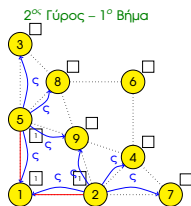
Οι διεργασίες 2, 5 θέτουν την 1 ως γονέα στο δέντρο.



## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncBFS

### 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 SyncBFS

### 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

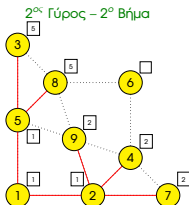
Η διεργασία 1 αγνοεί τα μηνύματα "αναζήτησης" που έλαβε.

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.

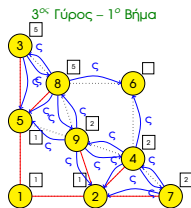
Η διεργασία 9 διαλέγει τυχαία την 2 ως γονέα στο δέντρο.



## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncBFS

### 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 SyncBFS

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

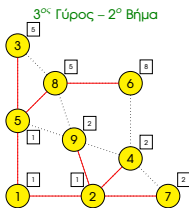
Οι διεργασίες **3, 4, 7, 8, 9** στέλνουν μήνυμα "αναζήτησης" σε όλους τους γείτονες τους.

### 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.

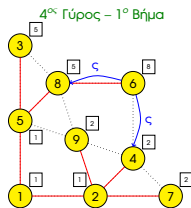
Η διεργασία **6** διαλέγει τυχαία την **8** ως γονέα στο δέντρο.



## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncBFS

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

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



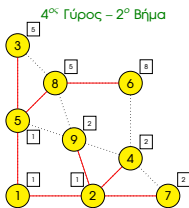
## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncBFS

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

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

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

Οι διεργασίες **4, 8** αγνοούν τα μηνύματα "αναζήτησης" που έλαβαν.

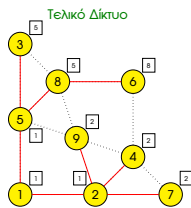


## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncBFS

### Τελικό Δίκτυο

Το δέντρο αναζήτησης κατά εύρος έχει κατασκευαστεί.

Ο αλγόριθμος εκτελέστηκε σε 4 γύρους. Συνολικά μεταδόθηκαν 28 μηνύματα.



## Τερματισμός (1)

Πώς μπορεί η διεργασία  $u_0$  να μάθει πότε ολοκληρώθηκε η κατασκευή του δέντρου;

- ▶ Δεν είναι γνωστή η διάμετρος του δικτύου και το πλήθος των διεργασιών  $n$

Οι διεργασίες απαντάνε στον αποστολέα ενός μηνύματος 'αναζήτησης' με ένα μήνυμα 'γονέας' ή 'μη-γονέας' ανάλογα του αν ο αποστολέας είναι ο γονέας της διεργασίας.

- ▶ Κάθε διεργασία που λαμβάνει ένα μήνυμα 'αναζήτησης' επιστρέφει στον αποστολέα **στον επόμενο γύρο** ένα μήνυμα 'μη-γονέας' αν ο αποστολέας δεν είναι ο γονέας της διεργασίας
- ▶ Η διεργασία καθυστερεί την αποστολή του μηνύματος 'γονέας' έως ότου λάβει κάποιο μήνυμα 'γονέας' ή 'μη-γονέας' από όλες τις διεργασίες στις οποίες έστειλε μηνύματα 'αναζήτησης'



## Τερματισμός (2)

Επομένως τα μηνύματα 'γονέας' ή 'μη-γονέας' έχουν δύο χρήσεις

1. **Παροχή πλήρους γνώσης** – κάθε διεργασία γνωρίζει και τα 'παιδιά' της
2. **Ενημέρωση τερματισμού** – κάθε διεργασία γνωρίζει πότε τα υποδέντρα των 'παιδιών' της έχουν κατασκευαστεί

Ο αλγόριθμος  $\text{SynchBFS}_T$  απαιτεί το πολύ  $2 \cdot \text{diam}(G) + 2$  γύρους και χρησιμοποιεί  $2 \cdot m$  μηνύματα



## Αλγόριθμος $\text{SynchBFS}_T$ (1)

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** – θέτει την μεταβλητή **μαρκαρισμένη** 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.

**true** – στον επόμενο γύρο, επιστρέφει στον αποστολέα ένα μήνυμα 'μη-γονέας'



## Αλγόριθμος $\text{SynchBFS}_T$ (2)

Όταν η διεργασία λάβει ένα μήνυμα 'μη-γονέας' από όλες τις γειτονικές διεργασίες, ή λάβει μήνυμα 'αναζήτησης' από την μοναδική γειτονική διεργασία, στον επόμενο γύρο στέλνει το μήνυμα 'γονέας' στον γονέα της.

Όταν η διεργασία λάβει ένα μήνυμα 'μη-γονέας' ή 'γονέας' από όλες τις γειτονικές διεργασίες, στον επόμενο γύρο στέλνει το μήνυμα 'γονέας' στον γονέα της.



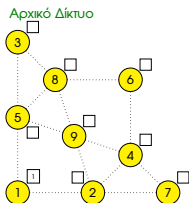
## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncBFS<sub>T</sub>

### Αρχικό Δίκτυο

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

Η διεργασία 1 θεωρείται μαρκιαρισμένη.

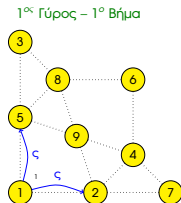
Όλες οι άλλες διεργασίες δεν είναι μαρκιαρισμένες.



## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncBFS<sub>T</sub>

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

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



## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncBFS<sub>T</sub>

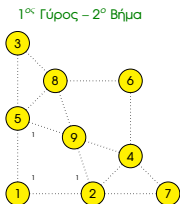
### 1<sup>ος</sup> Γύρος – 2<sup>ο</sup> Βήμα

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.

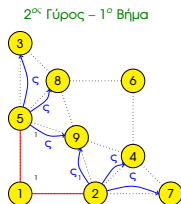
Οι διεργασίες 2, 5 θέτουν την 1 ως γονέα στο δέντρο.



## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncBFS<sub>T</sub>

### 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 SyncBFS<sub>T</sub>

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

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

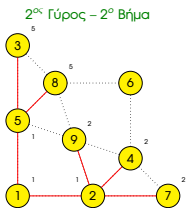
### 2<sup>ος</sup> Γύρος – 2<sup>ο</sup> Βήμα

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.



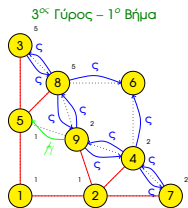
## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncBFS<sub>T</sub>

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

### 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.

Η διεργασία **9** στέλνει μήνυμα 'μη-γονέας' στην **5**



## Παράδειγμα Εκτέλεσης Αλγόριθμου SyncBFS<sub>T</sub>

### 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.

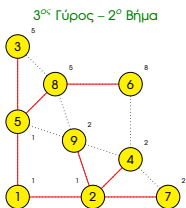
Η διεργασία **9** στέλνει μήνυμα 'μη-γονέας' στην **5**

### 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 SyncBFS<sub>T</sub>

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

Η διεργασία **3** στέλνει μήνυμα 'μη-γονέας' στην **8**

Η διεργασία **8** στέλνει μήνυμα 'μη-γονέας' στην **3**

Η διεργασία **8** στέλνει μήνυμα 'μη-γονέας' στην **9**

Η διεργασία **9** στέλνει μήνυμα 'μη-γονέας' στην **4**

Η διεργασία **4** στέλνει μήνυμα 'μη-γονέας' στην **7**

Η διεργασία **7** στέλνει μήνυμα 'μη-γονέας' στην **4**

Η διεργασία **4** στέλνει μήνυμα 'μη-γονέας' στην **4**

Η διεργασία **6** στέλνει μήνυμα 'γονέας' στην **8**

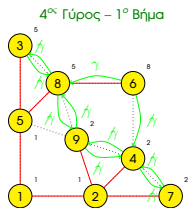
Η διεργασία **6** στέλνει μήνυμα 'γονέας' στην **8**

Η διεργασία **6** στέλνει μήνυμα 'γονέας' στην **8**

Η διεργασία **6** στέλνει μήνυμα 'γονέας' στην **8**

Η διεργασία **6** στέλνει μήνυμα 'γονέας' στην **8**

Η διεργασία **6** στέλνει μήνυμα 'γονέας' στην **8**





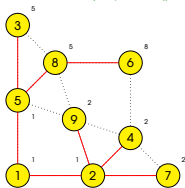
### 6<sup>ος</sup> Γύρος – 2<sup>ο</sup> Βήμα

**6<sup>ος</sup> Γύρος – 1<sup>ο</sup> Βήμα** Οι διεργασίες 2, 5 στέλνουν μήνυμα "γονέας" στην 1

**6<sup>ος</sup> Γύρος – 2<sup>ο</sup> Βήμα**

Η διεργασία 1 διαπιστώνει ότι η κατασκευή των υποδέντρων των 2, 5 ολοκληρώθηκε

Η διεργασία 1 τερματίζει

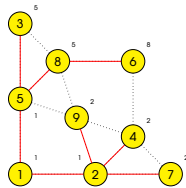


### Τελικό Δίκτυο

#### Τελικό Δίκτυο

Το δέντρο αναζήτησης κατά εύρος έχει κατασκευαστεί.

Ο αλγόριθμος εκτελέστηκε σε 6 γύρους.  
Συνολικά μεταδόθηκαν 36 μηνύματα.



## Εκφώνηση 1<sup>ου</sup> Προβλήματος

Θεωρείστε ένα σύγχρονο καταμετρημένο σύστημα με  $n$  διεργασίες συνδεδεμένες μέσω ενός γενικού, μη-κατευθυνόμενου δικτύου με  $m$  κανάλια επικοινωνίας. Κάθε διεργασία έχει μια μοναδική ταυτότητα και δεν γνωρίζει το σύνολο των διεργασιών, ούτε την τοπολογία του δικτύου. Σχεδιάστε έναν καταμετρημένο αλγόριθμο καταμέτρησης που επιτρέπει στη διεργασία  $u_i$  να υπολογίσει το πλήθος των διεργασιών. Αναλύστε την χρονική πολυπλοκότητα και πολυπλοκότητα μηνυμάτων. Αποδείξτε τους ισχυρισμούς σας.



## Καταγραφή Υποθέσεων / Μοντέλου

- ▶ Μοντέλο Επικοινωνίας
  - ▶ Σύγχρονο καταμετρημένο σύστημα
- ▶ Τοπολογία Δικτύου
  - ▶ Γενικό, μη-κατευθυνόμενο δίκτυο
  - ▶  $n$  διεργασίες,  $m$  κανάλια επικοινωνίας
- ▶ Σφάλματα
  - ▶ Κανένα
- ▶ Γνώσεις διεργασιών
  - ▶ Μοναδική ταυτότητα
  - ▶ Υπάρχει μια μοναδική διεργασία  $u_i$



- ▶ Κατανόηση Προβλήματος
  - ▶ Η διεργασία  $u_0$  πρέπει να υπολογίσει τη διάμετρο του δικτύου
  - ▶ Να μετρήσει τις διεργασίες
- ▶ Χαρακτηρισμός Προβλήματος
  - ▶ Κατασκευή ενός δένδρου αναζήτησης κατά εύρος με ρίζα την  $u_0$
  - ▶ Τα φύλλα ξεκινούν τους μετρίες.
  - ▶ Οι γονείς αθροίζουν τους μετρίες των παιδιών και στέλνουν το αποτέλεσμα στους γονείς τους.
- ▶ Αρχική Προσέγγιση
  - ▶ Υπάρχει κάποιος αντίστοιχος αλγόριθμος που μελετήσαμε στο μάθημα ;
  - ▶ Τι τροποποιήσεις πρέπει να κάνουμε ;



## Αλγόριθμος SynchBFS<sub>μ</sub> (1)

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** – θέτει την μεταβλητή **μαρκαρισμένη** 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.
- true** – στον επόμενο γύρο, επιστρέφει στον αποστολέα ένα μήνυμα 'μη-γονέας'



## Αλγόριθμος SynchBFS<sub>μ</sub> (2)

Όταν η διεργασία λάβει ένα μήνυμα 'μη-γονέας' από όλες τις γειτονικές διεργασίες, ή λάβει μήνυμα 'αναζήτησης' από την μοναδική γειτονική διεργασία, στον επόμενο γύρο στέλνει το μήνυμα (γονέας, 1) στον γονέα της.

Όταν η διεργασία λάβει ένα μήνυμα 'μη-γονέας' ή (γονέας, c) από όλες τις γειτονικές διεργασίες, αθροίζει τις μεταβλητές c όλων των μηνυμάτων 'γονέας' που έλαβε, αυξάνει το αποτέλεσμα κατά 1 και στον επόμενο γύρο στέλνει το μήνυμα (γονέας,  $\sum(c) + 1$ ) στον γονέα της.



- ▶ Ο αλγόριθμος SynchBFS<sub>μ</sub> κατασκευάζει ένα δέντρο αναζήτησης κατά εύρος όπου οι διεργασίες γνωρίζουν τον 'γονέα' τους στο δέντρο και τα 'παιδιά' τους.
- ▶ Ο αλγόριθμος SynchBFS<sub>μ</sub> κατασκευάζει ένα δέντρο αναζήτησης κατά εύρος όπου οι διεργασίες γνωρίζουν τον 'γονέα' τους στο δέντρο, τα 'παιδιά' τους και το πλήθος των διεργασιών που ανήκουν στα υποδέντρα των διεργασιών.
- ▶ Η χρονική πολυπλοκότητα είναι  $\mathcal{O}(diam(G))$
- ▶ Η πολυπλοκότητα επικοινωνίας είναι  $\mathcal{O}(m)$



### Αρχικό Δίκτυο

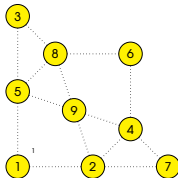
Το δίκτυο έχει 9 μονάδες, 14 κανάλια

Η διεργασία 1 ξεκινά την εκτέλεση του αλγορίθμου.

Η διεργασία 1 θεωρείται μαρκιαρισμένη.

Όλες οι άλλες διεργασίες δεν είναι μαρκιαρισμένες.

### Αρχικό Δίκτυο

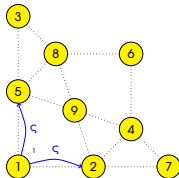


### 1<sup>ος</sup> Γύρος - 1<sup>ο</sup> Βήμα

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

Η διεργασία 1 στέλνει μήνυμα

'αναζήτησης' σε όλους του γείτονες της.



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

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

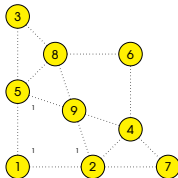
Η διεργασία 1 στέλνει μήνυμα

'αναζήτησης' σε όλους του γείτονες της.

#### 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.

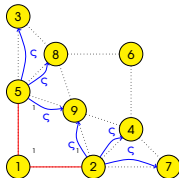


### 2<sup>ος</sup> Γύρος - 1<sup>ο</sup> Βήμα

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

Οι διεργασίες 2, 5 στέλνουν μήνυμα

'αναζήτησης' σε όλους τους γείτονες τους.



## 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

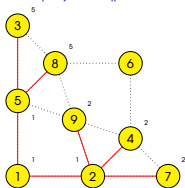
Οι διεργασίες **3, 4, 7, 8, 9** μαρκάρονται.

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.

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

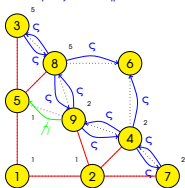


## 3<sup>ος</sup> Γύρος - 1<sup>ο</sup> Βήμα

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

Η διεργασία **9** στέλνει μήνυμα 'μη-γονέας' στην **5**

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



## 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.

Η διεργασία **9** στέλνει μήνυμα 'μη-γονέας' στην **5**

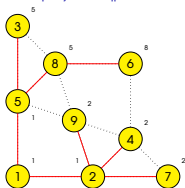
## 3<sup>ος</sup> Γύρος - 2<sup>ο</sup> Βήμα

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.

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



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

Η διεργασία **3** στέλνει μήνυμα 'μη-γονέας' στην **8**

Η διεργασία **8** στέλνει μήνυμα 'μη-γονέας' στην **3**

Η διεργασία **8** στέλνει μήνυμα 'μη-γονέας' στην **9**

Η διεργασία **9** στέλνει μήνυμα 'μη-γονέας' στην **4**

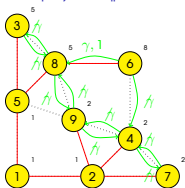
Η διεργασία **4** στέλνει μήνυμα 'μη-γονέας' στην **7**

Η διεργασία **7** στέλνει μήνυμα 'μη-γονέας' στην **4**

Η διεργασία **6** στέλνει μήνυμα 'μη-γονέας' στην **4**

Η διεργασία **6** στέλνει μήνυμα 'γονέας,1' στην **8**

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



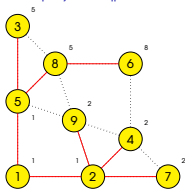
#### 4<sup>ος</sup> Γύρος – 1<sup>ο</sup> Βήμα

Η διεργασία 3 στέλνει μήνυμα 'μη-γονέας' στην 8  
Η διεργασία 8 στέλνει μήνυμα 'μη-γονέας' στην 3  
Η διεργασία 8 στέλνει μήνυμα 'μη-γονέας' στην 9  
Η διεργασία 9 στέλνει μήνυμα 'μη-γονέας' στην 4  
Η διεργασία 4 στέλνει μήνυμα 'μη-γονέας' στην 7  
Η διεργασία 7 στέλνει μήνυμα 'μη-γονέας' στην 4  
Η διεργασία 6 στέλνει μήνυμα 'μη-γονέας' στην 4  
Η διεργασία 6 στέλνει μήνυμα 'γονέας,1' στην 8

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

Η διεργασία 8 διαπιστώνει ότι η κατασκευή του υποδέντρου της 6 ολοκληρώθηκε

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



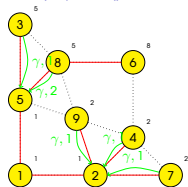
#### 5<sup>ος</sup> Γύρος – 1<sup>ο</sup> Βήμα

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

Η διεργασία 8 στέλνει μήνυμα 'γονέας,2' στην 5

Η διεργασία 3 στέλνει μήνυμα 'γονέας,1' στην 5

Οι διεργασίες 4, 7, 9 στέλνουν μήνυμα 'γονέας,1' στην 2



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

Η διεργασία 8 στέλνει μήνυμα 'γονέας,2' στην 5

Η διεργασία 3 στέλνει μήνυμα 'γονέας,1' στην 5

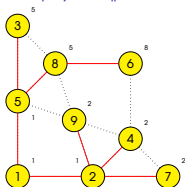
Οι διεργασίες 4, 7, 9 στέλνουν μήνυμα 'γονέας,1' στην 2

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

Η διεργασία 5 διαπιστώνει ότι η κατασκευή των υποδέντρων των 3, 8 ολοκληρώθηκε

Η διεργασία 2 διαπιστώνει ότι η κατασκευή των υποδέντρων των 4, 7, 9 ολοκληρώθηκε

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

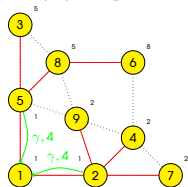


#### 6<sup>ος</sup> Γύρος – 1<sup>ο</sup> Βήμα

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

Η διεργασία 2 στέλνει μήνυμα 'γονέας,4' στην 1

Η διεργασία 5 στέλνει μήνυμα 'γονέας,4' στην 1



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

Η διεργασία 2 στέλνει μήνυμα 'γονέας,4' στην 1

Η διεργασία 5 στέλνει μήνυμα 'γονέας,4' στην 1

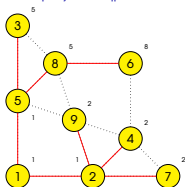
### 6<sup>ος</sup> Γύρος – 2<sup>ο</sup> Βήμα

Η διεργασία 1 διαπιστώνει ότι η κατασκευή των υποδέντρων των 2, 5 ολοκληρώθηκε

Η διεργασία 1 πλέον γνωρίζει το σύνολο των διεργασιών: 9

Η διεργασία 1 τερματίζει

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



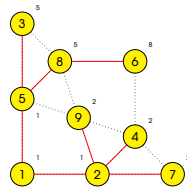
### Τελικό Δίκτυο

#### Τελικό Δίκτυο

Η καταμέτρηση των διεργασιών έχει ολοκληρωθεί.

Ο αλγόριθμος εκτελέστηκε σε 6 γύρους.

Συνολικά μεταδόθηκαν 36 μηνύματα.



## Εκφώνηση 2<sup>ου</sup> Προβλήματος

Θεωρείστε ένα σύγχρονο κατακευθμισμένο σύστημα με  $n$  διεργασίες συνδεδεμένες μέσω ενός γενικού, μη-κατευθυνόμενου δικτύου με  $m$  κανάλια επικοινωνίας. Κάθε διεργασία έχει μια μοναδική ταυτότητα και δεν γνωρίζει το σύνολο των διεργασιών, ούτε την τοπολογία του δικτύου. Σχεδιάστε έναν κατακευθμισμένο αλγόριθμο που επιτρέπει στη διεργασία  $u_0$  να υπολογίσει τη διάμετρο του δικτύου. Αναλύστε την χρονική πολυπλοκότητα και πολυπλοκότητα μηνυμάτων. Αποδείξτε τους ισχυρισμούς σας.



## Καταγραφή Υποθέσεων / Μοντέλου

- ▶ Μοντέλο Επικοινωνίας
  - ▶ Σύγχρονο κατακευθμισμένο σύστημα
- ▶ Τοπολογία Δικτύου
  - ▶ Γενικό, μη-κατευθυνόμενο δίκτυο
  - ▶  $n$  διεργασίες,  $m$  κανάλια επικοινωνίας
- ▶ Σφάλματα
  - ▶ Κανένα
- ▶ Γνώσεις διεργασιών
  - ▶ Μοναδική ταυτότητα
  - ▶ Υπάρχει μια μοναδική διεργασία  $u_0$



- ▶ Κατανόηση Προβλήματος
  - ▶ Η διεργασία  $u_3$  πρέπει να υπολογίσει τη διάμετρο του δικτύου
  - ▶ Να βρει το ελάχιστο μέγιστο μονοπάτι
- ▶ Χαρακτηρισμός Προβλήματος
  - ▶ Κατασκευή ενός δένδρου αναζήτησης κατά εύρος με ρίζα την  $u_3$
  - ▶ Υπολογίζει όλα τα ελάχιστα μονοπάτια από την  $u_3$  προς όλες τις άλλες διεργασίες
  - ▶ Υπολογίζει το μήκος όλων των μονοπατιών
  - ▶ Εντοπίζει το μέγιστο μήκος – αυτή είναι η διάμετρος
- ▶ Αρχική Προσέγγιση
  - ▶ Υπάρχει κάποιος αντίστοιχος αλγόριθμος που μελετήσαμε στο μάθημα ;
  - ▶ Τι τροποποιήσεις πρέπει να κάνουμε ;



Θεωρείστε ένα σύγχρονο κατανεμημένο σύστημα με  $n$  διεργασίες συνδεδεμένες μέσω ενός γενικού, μη-κατευθυνόμενου δικτύου με  $m$  κανάλια επικοινωνίας, όπου κάθε διεργασία έχει μια μοναδική ταυτότητα αλλά δεν γνωρίζει το σύνολο των διεργασιών, ούτε την τοπολογία του δικτύου. Κάθε διεργασία  $u$  δέχεται ως είσοδο έναν ακέραιο αριθμό  $l_u$ . Σχεδιάστε έναν κατανεμημένο αλγόριθμο που επιτρέπει σε όλες τις διεργασίες να υπολογίσουν τον μέσο όρο όλων των αριθμών από αυτούς που έχουν δοθεί ( $\sum_{u=1}^n l_u/n$ ). Αναλύστε την χρονική πολυπλοκότητα και πολυπλοκότητα μηνυμάτων. Αποδείξτε τους ισχυρισμούς σας.



- ▶ Μοντέλο Επικοινωνίας
  - ▶ Σύγχρονο κατανεμημένο σύστημα
- ▶ Τοπολογία Δικτύου
  - ▶ Γενικό, μη-κατευθυνόμενο δίκτυο
  - ▶  $n$  διεργασίες,  $m$  κανάλια επικοινωνίας
- ▶ Σφάλματα
  - ▶ Κανένα
- ▶ Γνώσεις διεργασιών
  - ▶ Μοναδική ταυτότητα
  - ▶ έναν ακέραιο αριθμό  $l_u$



- ▶ Κατανόηση Προβλήματος
  - ▶ Όλες οι διεργασίες να υπολογίσουν τον μέσο όρο όλων των αριθμών από αυτούς που έχουν δοθεί ( $\sum_{u=1}^n l_u/n$ )
- ▶ Χαρακτηρισμός Προβλήματος
  - ▶ Πρέπει οι διεργασίες να 'μάθουν' τους αριθμούς όλων των άλλων διεργασιών
  - ▶ Πρέπει οι διεργασίες να 'μάθουν' το πλήθος των διεργασιών
  - ▶ Κατασκευή ενός δένδρου αναζήτησης
- ▶ Αρχική Προσέγγιση
  - ▶ Όλες οι διεργασίες παράλληλα ή εκλέγουμε μια συγκεκριμένη ;
  - ▶ Υπάρχει κάποιος αντίστοιχος αλγόριθμος που μελετήσαμε στο μάθημα ;
  - ▶ Τι τροποποιήσεις πρέπει να κάνουμε ;
  - ▶ Συνδυασμός 1+ αλγόριθμων ;



## Εκφώνηση 4<sup>ου</sup> Προβλήματος

Σχεδιάστε έναν αλγόριθμο για σύγχρονα καταμεμημένα συστήματα, όπου το δίκτυο επικοινωνίας μπορεί να αναπαρασταθεί από ένα γενικό γράφημα (μη-κατευθυνόμενο), που επιτρέπει σε μία διεργασία  $u_i$  να εντοπίσει την διεργασία  $u_{max}$  που έχει την μεγαλύτερη απόσταση στο γράφημα (δηλ. να μάθει την ταυτότητά της  $u_{max}$  και την απόσταση της). Αναλύστε την χρονική πολυπλοκότητα και πολυπλοκότητα μηνυμάτων. Αποδείξτε τους ισχυρισμούς σας.



## Εκφώνηση 5<sup>ου</sup> Προβλήματος

Σχεδιάστε έναν αλγόριθμο για το πρόβλημα της αναζήτησης κατά βάθος σε σύγχρονα καταμεμημένα συστήματα, όπου το δίκτυο επικοινωνίας μπορεί να αναπαρασταθεί από ένα γενικό γράφημα (μη-κατευθυνόμενο). Υποθέτουμε μια διεργασία  $u_i$  η οποία ξεκινάει την εκτέλεση του αλγορίθμου. Στο τέλος της εκτέλεσης του αλγορίθμου όλες οι διεργασίες γνωρίζουν τον γονέα τους στο δέντρο αναζήτησης κατά βάθος. Αναλύστε την χρονική πολυπλοκότητα και πολυπλοκότητα μηνυμάτων. Αποδείξτε τους ισχυρισμούς σας.



## Εκφώνηση 6<sup>ου</sup> Προβλήματος

Θεωρείστε ένα σύγχρονο καταμεμημένο σύστημα με  $n$  διεργασίες συνδεδεμένες μέσω ενός γενικού, μη-κατευθυνόμενου δικτύου με  $m$  κανάλια επικοινωνίας, όπου κάθε διεργασία δεν γνωρίζει το σύνολο των διεργασιών, ούτε την τοπολογία του δικτύου. Κάθε διεργασία  $u_i$  δέχεται ως είσοδο έναν ακέραιο αριθμό  $i_{in}$ . Σχεδιάστε έναν καταμεμημένο αλγόριθμο που επιτρέπει στη διεργασία  $u_i$  να υπολογίσει τον μέγιστο αριθμό από αυτούς που έχουν δοθεί στις διεργασίες. Αναλύστε την χρονική πολυπλοκότητα και πολυπλοκότητα μηνυμάτων.



## Εκφώνηση 7<sup>ου</sup> Προβλήματος

Σχεδιάστε έναν αλγόριθμο για το πρόβλημα της κατασκευής ενός εικονικού δακτυλίου σε σύγχρονα καταμεμημένα συστήματα, όπου το δίκτυο επικοινωνίας μπορεί να αναπαρασταθεί από ένα γενικό γράφημα (μη-κατευθυνόμενο). Υποθέτουμε μια διεργασία  $u_i$  η οποία ξεκινάει την εκτέλεση του αλγορίθμου. Στο τέλος της εκτέλεσης του αλγορίθμου όλες οι διεργασίες γνωρίζουν τους γειτονικούς κόμβους στο εικονικό δακτύλιο. Αναλύστε την χρονική πολυπλοκότητα και πολυπλοκότητα μηνυμάτων. Αποδείξτε τους ισχυρισμούς σας.



## Εκφώνηση 8<sup>ου</sup> Προβλήματος

Σχεδιάστε έναν αλγόριθμο καταμέτρησης των διεργασιών για σύγχρονα καταναμημένα συστήματα, όπου το δίκτυο επικοινωνίας μπορεί να αναπαρασταθεί από ένα γενικό γράφημα (μη-κατευθυνόμενο). Κάθε διεργασία έχει μια μοναδική ταυτότητα, δεν γνωρίζει την διάμετρο του δικτύου, και δεν γνωρίζει το συνολικό αριθμό διεργασιών  $n$ . Αναλύστε την χρονική πολυπλοκότητα και πολυπλοκότητα μηνυμάτων. Αποδείξτε τους ισχυρισμούς σας.



## Εκφώνηση 9<sup>ου</sup> Προβλήματος

Έστω ένα σύγχρονο δίκτυο δακτυλίου από  $n$  διεργασίες, όπου κάθε διεργασία έχει μια μοναδική ταυτότητα και δεν γνωρίζει το σύνολο των διεργασιών. Σχεδιάστε έναν καταναμημένο αλγόριθμο που να λύνει το 'πρόβλημα της αναγνώρισης': κάθε διεργασία είναι ενήμερη για τις ταυτότητες όλων των άλλων διεργασιών. Αναλύστε την χρονική πολυπλοκότητα και πολυπλοκότητα μηνυμάτων. Αποδείξτε τους ισχυρισμούς σας.



## Εκφώνηση 10<sup>ου</sup> Προβλήματος

Θεωρείστε ένα σύγχρονο καταναμημένο σύστημα με  $n$  διεργασίες συνδεδεμένες μέσω ενός γενικού δικτύου. Κάθε διεργασία έχει μια μοναδική ταυτότητα και δεν γνωρίζει το σύνολο των διεργασιών ή την τοπολογία του δικτύου. Σχεδιάστε τον πιο αποδοτικό αλγόριθμο που μπορείτε για την κατασκευή ενός ελάχιστου-ύψους επικαλυπτικού δέντρου. Ορίστε προσεκτικά τις ιδιότητες του αλγόριθμου σας και αναλύστε την χρονική πολυπλοκότητα και πολυπλοκότητα μηνυμάτων.



Περισσότερα προβλήματα στο site του μαθήματος !

