

Κατανεμημένα Συστήματα I

Μάθημα Βασικής Επιλογής, Χειμερινού Εξαμήνου

Τομέας Εφαρμογών και Θεμελιώσεων

Ιωάννης Χατζηγιαννάκης

Δευτέρα, 31 Οκτωβρίου, 2011
Αίθουσα Β3



Προηγούμενο Μάθημα

- ▶ Σύγχρονα Κατανεμημένα Συστήματα
- ▶ Μοντελοποίηση Συστήματος
- ▶ Πρόβλημα Εκλογής Αρχηγού
- ▶ Μελέτη ενός κατανεμημένου αλγόριθμου 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^ο Βήμα

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

2^ο 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.



Πρόβλημα Εκλογής Αρχηγού

- ▶ Ορισμός Προβλήματος
- ▶ Αδυναμία αντιμετώπισης του προβλήματος όταν οι διεργασίες δεν έχουν μοναδικές ταυτότητες
- ▶ Δίκτυα Δακτυλίου
 1. Αλγόριθμος των LeLann, Chang και Roberts
 - ▶ Η χρονική πολυπλοκότητα είναι $\mathcal{O}(n)$
 - ▶ Η πολυπλοκότητα επικοινωνίας είναι $\mathcal{O}(n^2)$
- ▶ Γενικά Δίκτυα
 - ▶ Αλγόριθμος FloodMax
 - ▶ Η χρονική πολυπλοκότητα είναι $\mathcal{O}(\text{diam}(G))$
 - ▶ Η πολυπλοκότητα επικοινωνίας είναι $\mathcal{O}(\text{diam}(G) \cdot m)$



Ορισμός Προβλήματος

Ένας αλγόριθμος λύνει το πρόβλημα εκλογής αρχηγού εφόσον πληροί τις παρακάτω προδιαγραφές:

- ▶ Όλες οι καταστάσεις τερματισμού χωρίζονται σε δύο σύνολα:
 1. καταστάσεις που αναδεικνύουν την διεργασία ως `εκλεγμένη`
 2. καταστάσεις που θεωρούν την διεργασία ως `μη εκλεγμένη`
- ▶ Όταν μια διεργασία εισέρθει σε μία κατάσταση τερματισμού, η συνάρτηση αλλαγής κατάστασης θα μεταβεί μόνο σε κάποια κατάσταση του ίδιου συνόλου
- ▶ Σε κάθε εκτέλεση του αλγορίθμου:
 - ▶ μία και μόνο μία διεργασία είναι `εκλεγμένη`
 - ▶ οι άλλες διεργασίες βρίσκονται σε κατάσταση `μη-εκλεγμένη`



Αδυναμία αντιμετώπισης του προβλήματος (1)

Παρατηρούμε ότι εάν οι διεργασίες δεν μπορούν να διαχωριστούν μεταξύ τους, τότε δεν μπορεί να βρεθεί κάποιος αλγόριθμος που να δίνει λύση στο πρόβλημα

Θεώρημα

Έστω ένα σύστημα \mathcal{A} από n διεργασίες συνδεδεμένες μέσω ενός δικτύου δακτυλίου. Αν όλες οι διεργασίες είναι πανομοιότυπες και δεν μπορούν να ξεχωρίσουν η μία την άλλη, τότε κανένας αλγόριθμος δεν μπορεί να λύσει το πρόβλημα εκλογής αρχηγού για το σύστημα \mathcal{A} .

- ▶ Το θεώρημα ισχύει ακόμα και όταν ο συνολικός αριθμός των διεργασιών n είναι γνωστός σε κάθε διεργασία
- ▶ ή το δίκτυο έχει συγκεκριμένες ιδιότητες, π.χ. τα κανάλια είναι μονής ή διπλής κατεύθυνσης.



Αδυναμία αντιμετώπισης του προβλήματος (2)

Απόδειξη: Χρησιμοποιούμε την μέθοδο της εις άτοπον απαγωγής

- ▶ Έστω αλγόριθμος που λύνει το πρόβλημα για το \mathcal{A}
- ▶ Έστω ότι κάθε διεργασία έχει μόνο μια αρχική κατάσταση
- ▶ Άρα όλες οι διεργασίες ξεκινούν από την ίδια κατάσταση

Επαγωγικά: τον γύρο r όλες οι διεργασίες θα βρίσκονται στην ίδια κατάσταση

- ▶ αν κάποια διεργασία βρεθεί σε κατάσταση **εκλεγμένη**
- ▶ τότε όλες οι άλλες διεργασίες θα έχουν βρεθεί σε μια ίδια κατάσταση
- ▶ δεν υπάρχει μια και μόνο μια αρχηγός



Ο Αλγόριθμος των LeLann, Chang και Roberts

Αλγόριθμος LCR

Οι διεργασίες διατηρούν μια μεταβλητή **αρχηγός** η οποία αρχικά είναι `false`. Οι διεργασίες εκπέμπουν την ταυτότητα τους στον δεξιόστροφο γείτονα τους. Μόλις λάβουν μία ταυτότητα από τον αριστερόστροφο γείτονα, την συγκρίνουν με την δικιά τους. Αν είναι μεγαλύτερη, την προωθούν στον δεξιόστροφο γείτονα. Αν είναι μικρότερη, δεν κάνουν τίποτα. Αν είναι ίδια, μεταβαίνουν στην κατάσταση **εκλεγμένη** θέτοντας την μεταβλητή **αρχηγός** στην τιμή `true`.

- ▶ Δεν γνωρίζει τον συνολικό αριθμό των διεργασιών
- ▶ Υποθέτει ότι οι διεργασίες μπορούν να επικοινωνήσουν μόνο προς μία κατεύθυνση – δεξιόστροφα
- ▶ Βασίζεται σε απλές πράξεις σύγκρισης ταυτοτήτων
- ▶ Αποκεντρωτικός αλγόριθμος

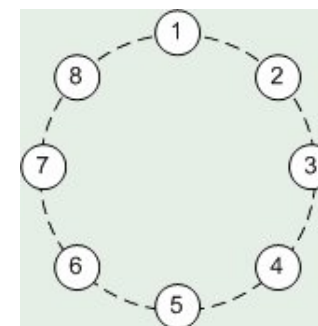


Παράδειγμα Εκτέλεσης Αλγόριθμου LCR

▶ Έστω ένα σύγχρονο καταναμημένο σύστημα από $n = 8$ διεργασίες.

- ▶ Δίκτυο δακτυλίου
- ▶ Οι διεργασίες είναι αριθμημένες από 1 έως 8 δεξιόστροφα

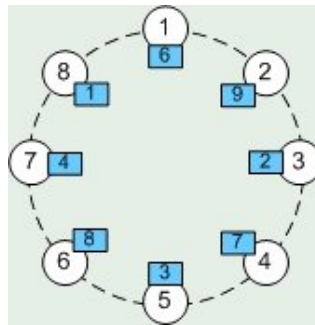
Διεργασίες σε δακτύλιο



Παράδειγμα Εκτέλεσης Αλγόριθμου LCR

- ▶ Έστω ένα σύγχρονο καταναμημένο σύστημα από $n = 8$ διεργασίες.
 - ▶ Δίκτυο δακτυλίου
 - ▶ Οι διεργασίες είναι αριθμημένες από 1 έως 8 δεξιόστροφα
- ▶ Οι διεργασίες έχουν μοναδικές ταυτότητες
 - ▶ Δεν γνωρίζουν την ταυτότητα των υπόλοιπων διεργασιών

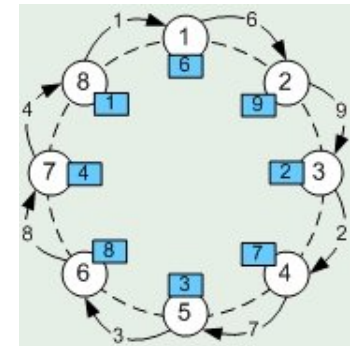
Διεργασίες σε δακτύλιο



Παράδειγμα Εκτέλεσης Αλγόριθμου LCR

- ▶ Έστω ένα σύγχρονο καταναμημένο σύστημα από $n = 8$ διεργασίες.
 - ▶ Δίκτυο δακτυλίου
 - ▶ Οι διεργασίες είναι αριθμημένες από 1 έως 8 δεξιόστροφα
- ▶ Οι διεργασίες έχουν μοναδικές ταυτότητες
 - ▶ Δεν γνωρίζουν την ταυτότητα των υπόλοιπων διεργασιών
- ▶ Πρώτος Γύρος – αποστολή μηνυμάτων

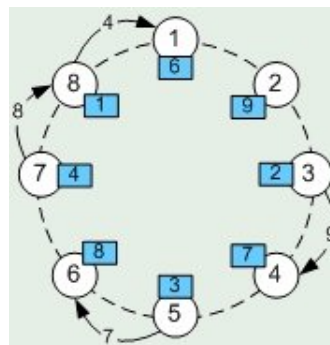
Διεργασίες σε δακτύλιο



Παράδειγμα Εκτέλεσης Αλγόριθμου LCR

- ▶ Έστω ένα σύγχρονο καταναμημένο σύστημα από $n = 8$ διεργασίες.
 - ▶ Δίκτυο δακτυλίου
 - ▶ Οι διεργασίες είναι αριθμημένες από 1 έως 8 δεξιόστροφα
- ▶ Οι διεργασίες έχουν μοναδικές ταυτότητες
 - ▶ Δεν γνωρίζουν την ταυτότητα των υπόλοιπων διεργασιών
- ▶ Πρώτος Γύρος – αποστολή μηνυμάτων
- ▶ Δεύτερος Γύρος

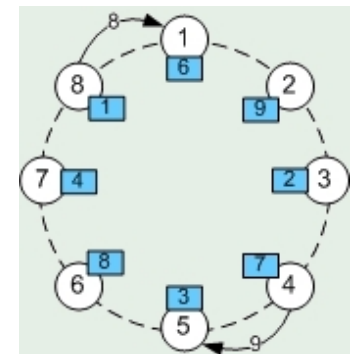
Διεργασίες σε δακτύλιο



Παράδειγμα Εκτέλεσης Αλγόριθμου LCR

- ▶ Έστω ένα σύγχρονο καταναμημένο σύστημα από $n = 8$ διεργασίες.
 - ▶ Δίκτυο δακτυλίου
 - ▶ Οι διεργασίες είναι αριθμημένες από 1 έως 8 δεξιόστροφα
- ▶ Οι διεργασίες έχουν μοναδικές ταυτότητες
 - ▶ Δεν γνωρίζουν την ταυτότητα των υπόλοιπων διεργασιών
- ▶ Πρώτος Γύρος – αποστολή μηνυμάτων
- ▶ Δεύτερος Γύρος
- ▶ Επόμενοι Γύροι

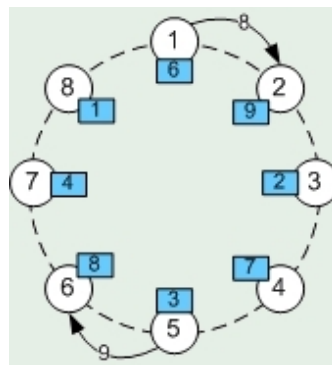
Διεργασίες σε δακτύλιο



Παράδειγμα Εκτέλεσης Αλγόριθμου LCR

- ▶ Έστω ένα σύγχρονο καταναμημένο σύστημα από $n = 8$ διεργασίες.
 - ▶ Δίκτυο δακτυλίου
 - ▶ Οι διεργασίες είναι αριθμημένες από 1 έως 8 δεξιόστροφα
- ▶ Οι διεργασίες έχουν μοναδικές ταυτότητες
 - ▶ Δεν γνωρίζουν την ταυτότητα των υπόλοιπων διεργασιών
- ▶ Πρώτος Γύρος – αποστολή μηνυμάτων
- ▶ Δεύτερος Γύρος
- ▶ Επόμενοι Γύροι

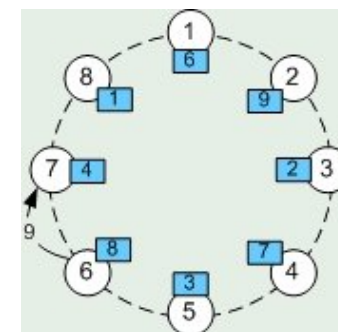
Διεργασίες σε δακτύλιο



Παράδειγμα Εκτέλεσης Αλγόριθμου LCR

- ▶ Έστω ένα σύγχρονο καταναμημένο σύστημα από $n = 8$ διεργασίες.
 - ▶ Δίκτυο δακτυλίου
 - ▶ Οι διεργασίες είναι αριθμημένες από 1 έως 8 δεξιόστροφα
- ▶ Οι διεργασίες έχουν μοναδικές ταυτότητες
 - ▶ Δεν γνωρίζουν την ταυτότητα των υπόλοιπων διεργασιών
- ▶ Πρώτος Γύρος – αποστολή μηνυμάτων
- ▶ Δεύτερος Γύρος
- ▶ Επόμενοι Γύροι

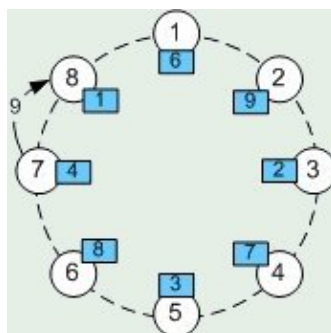
Διεργασίες σε δακτύλιο



Παράδειγμα Εκτέλεσης Αλγόριθμου LCR

- ▶ Έστω ένα σύγχρονο καταναμημένο σύστημα από $n = 8$ διεργασίες.
 - ▶ Δίκτυο δακτυλίου
 - ▶ Οι διεργασίες είναι αριθμημένες από 1 έως 8 δεξιόστροφα
- ▶ Οι διεργασίες έχουν μοναδικές ταυτότητες
 - ▶ Δεν γνωρίζουν την ταυτότητα των υπόλοιπων διεργασιών
- ▶ Πρώτος Γύρος – αποστολή μηνυμάτων
- ▶ Δεύτερος Γύρος
- ▶ Επόμενοι Γύροι

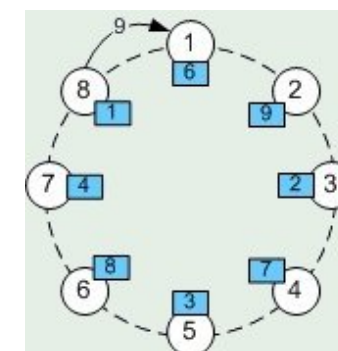
Διεργασίες σε δακτύλιο



Παράδειγμα Εκτέλεσης Αλγόριθμου LCR

- ▶ Έστω ένα σύγχρονο καταναμημένο σύστημα από $n = 8$ διεργασίες.
 - ▶ Δίκτυο δακτυλίου
 - ▶ Οι διεργασίες είναι αριθμημένες από 1 έως 8 δεξιόστροφα
- ▶ Οι διεργασίες έχουν μοναδικές ταυτότητες
 - ▶ Δεν γνωρίζουν την ταυτότητα των υπόλοιπων διεργασιών
- ▶ Πρώτος Γύρος – αποστολή μηνυμάτων
- ▶ Δεύτερος Γύρος
- ▶ Επόμενοι Γύροι

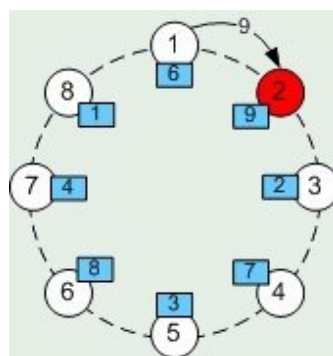
Διεργασίες σε δακτύλιο



Παράδειγμα Εκτέλεσης Αλγόριθμου LCR

- ▶ Έστω ένα σύγχρονο καταναμημένο σύστημα από $n = 8$ διεργασίες.
 - ▶ Δίκτυο δακτυλίου
 - ▶ Οι διεργασίες είναι αριθμημένες από 1 έως 8 δεξιόστροφα
- ▶ Οι διεργασίες έχουν μοναδικές ταυτότητες
 - ▶ Δεν γνωρίζουν την ταυτότητα των υπόλοιπων διεργασιών
- ▶ Πρώτος Γύρος – αποστολή μηνυμάτων
- ▶ Δεύτερος Γύρος
- ▶ Επόμενοι Γύροι
- ▶ **Εκλογή αρχηγού – διεργασία 2**

Διεργασίες σε δακτύλιο



Σύνοψη 3^{ης} Διάλεξης

Προηγούμενο Μάθημα

Προηγούμενο Μάθημα
Σύγχρονα Καταναμημένα Συστήματα
Εκλογή Αρχηγού

Πρόβλημα Εκλογής Αρχηγού

Καταναμημένοι Αλγόριθμοι σε Δίκτυα Δακτυλίου
Ανώνυμα Δίκτυα
Καταναμημένοι Αλγόριθμοι σε Γενικά Δίκτυα

Σύνοψη Μαθήματος

Σύνοψη Μαθήματος
Βιβλιογραφία
Επόμενο Μάθημα

Ο Αλγόριθμος TimeSlice

Αλγόριθμος TimeSlice

Οι διεργασίες διατηρούν μια μεταβλητή **αρχηγός** η οποία αρχικά είναι `false`, και ένα **μετρητή γύρων** l , αρχικά 0. Κάθε διεργασία u λειτουργεί σε φάσεις $(0, 1, 2, \dots)$, όπου κάθε φάση διαρκεί n γύρους. Κάθε φάση v (που αποτελείται από τους γύρους $(v-1)n+1, \dots, vn$) αποσκοπεί στην περιφορά του μήνυματος που περιέχει την ταυτότητα της διεργασίας v .

Αν η διεργασία i υπάρχει, και μέχρι τον γύρο $(i-1)n+1$ δεν έχει λάβει κάποιο μήνυμα, θέτει την μεταβλητή **αρχηγός** σε `true` και στέλνει ένα μήνυμα με την ταυτότητα της στον δεξιόστροφο γείτονα της. Κάθε διεργασία που λαμβάνει το μήνυμα, στέλνει το μήνυμα στον δεξιόστροφο γείτονα της και τερματίζει.

- ▶ Βασίζεται στην γνώση του πλήθους των διεργασιών n

Χαρακτηριστικά του Αλγόριθμου TimeSlice

- ▶ Ο αλγόριθμος εκλέγει την διεργασία με την μικρότερη ταυτότητα i_{min}
- ▶ Καμία διεργασία εκτός της i_{min} δεν είναι σε κατάσταση 'εκλεγμένη'
- ▶ Η διεργασία i_{min} εκλέγεται αρχηγός στο γύρο $(i_{min}-1)n+1$
- ▶ Μέχρι τον γύρο $(i_{min}-1)n+1$ και μετά τον γύρο $i_{min}n$ κανένα μήνυμα δεν ανταλλάσσεται
- ▶ Ο συνολικός αριθμός μηνυμάτων είναι n
- ▶ Η πολυπλοκότητα επικοινωνίας είναι $\mathcal{O}(n)$
- ▶ Η χρονική πολυπλοκότητα είναι $n i_{min}$ -- δεν φράσσεται ακόμα και σε δακτυλίου σταθερού μεγέθους.

Αλγόριθμος HS

Οι διεργασίες διατηρούν μια μεταβλητή **αρχηγός** η οποία αρχικά είναι `false`, και ένα **μετρητή φάσης** l , αρχικά 0. Κάθε διεργασία u λειτουργεί σε φάσεις $(0, 1, 2, \dots)$. Στην αρχή κάθε φάσης l , η διεργασία u στέλνει μηνύματα που περιέχουν την ταυτότητα της και ένα μετρητή προς τους δύο γείτονές της. Ο μετρητής στα μηνύματα τίθεται στην τιμή 2^l . Αν επιστρέψουν και τα δύο αντίγραφα τότε η u συνεχίζει με τη φάση $l + 1$. Όταν μια διεργασία λάβει ένα μήνυμα, συγκρίνει την ταυτότητα του μηνύματος με την δικιά τους. Αν είναι μεγαλύτερη, την προωθεί στον επόμενο γείτονα. Αν είναι μικρότερη, δεν κάνει τίποτα. Αν είναι ίδια και ο μετρητής είναι θετικός, μεταβαίνει στην κατάσταση **εκλεγμένη** θέτοντας την μεταβλητή **αρχηγός** στην τιμή `true`.



Θεώρημα

Ο αλγόριθμος HS ανταλλάσει $\mathcal{O}(n \log n)$ μηνύματα.

Απόδειξη: Στη φάση 0 κάθε διεργασία στέλνει την ταυτότητα της προς τις δύο κατευθύνσεις σε απόσταση 1. Εφόσον τα μηνύματα δεν καταστραφούν, επιστρέφουν πίσω στην διεργασία. Άρα στη φάση 0 ανταλλάσσονται το πολύ $4n$ μηνύματα.

Σε κάποια φάση $l > 0$ μια διεργασία στέλνει την ταυτότητα της αν και τα δύο μηνύματα επέστρεψαν στη φάση $l - 1$, δηλαδή καμία άλλη διεργασία σε απόσταση το πολύ 2^{l-1} δεν είχε μεγαλύτερη ταυτότητα. Αυτό συνεπάγεται ότι σε οποιαδήποτε ομάδα από $2^{l-1} + 1$ συνεχόμενες διεργασίες το πολύ μία θα στείλει την ταυτότητα της στη φάση l .



Πολυπλοκότητα Επικοινωνίας (2)

Συνεπώς το πολύ

$$\left\lfloor \frac{n}{2^{l-1} + 1} \right\rfloor$$

διεργασίες θα στείλουν την ταυτότητα τους στη φάση l . Άρα στη φάση l ανταλλάσσονται το πολύ

$$4 \left(\left\lfloor \frac{n}{2^{l-1} + 1} \right\rfloor \times 2^l \right) \leq 8n$$

μηνύματα.

Ο συνολικός αριθμός των φάσεων είναι $1 + \lceil \log n \rceil$, οπότε ο συνολικός αριθμός μηνυμάτων είναι μικρότερος από $8n(1 + \lceil \log n \rceil)$.



Χρονική Πολυπλοκότητα

Θεώρημα

Ο αλγόριθμος HS απαιτεί $\mathcal{O}(n)$ γύρους.

Απόδειξη: Για κάθε φάση l απαιτούνται 2^{l+1} γύροι για τις ταυτότητες να διασχύσουν όλο το δακτύλιο και να επιστρέψουν. Εξαιρείται η τελευταία φάση που χρειάζεται ακριβώς n γύρους μέχρι την ανακήρυξη του αρχηγού. Άρα ο συνολικός αριθμός γύρων που απαιτούνται για τον HS είναι

$$\sum_{l=0}^{\lceil \log n \rceil - 1} 2^{l+1} + n = 2^{\lceil \log n \rceil + 1} - 2 + n < 5n$$



Ανώνυμα Δίκτυα – Ύπαρξη συμμετρίας στο δίκτυο

- ▶ Επανεξετάζουμε την περίπτωση όπου οι διεργασίες δεν έχουν ταυτότητες
- ▶ Διακρίνουμε την ύπαρξη μιας συμμετρίας στο δίκτυο
- ▶ Αναφέρεται επίσης ως το **πρόβλημα διάσπασης της συμμετρίας**
- ▶ Ο αλγόριθμος των Itai & Rodeh λύνει το πρόβλημα
 - ▶ Βασίζεται στον αλγόριθμο LCR
 - ▶ **Βασική διαφορά:** οι διεργασίες δεν έχουν μοναδική ταυτότητα
- ▶ Κάθε διεργασία διαλέγει τυχαία μια ταυτότητα από το σύνολο $\{1, \dots, n\}$
- ▶ Ψάχνουμε να βρούμε την διεργασία με την μεγαλύτερη ταυτότητα
 - ▶ Αν διαπιστώσουμε ότι η μεγαλύτερη ταυτότητα δεν είναι μοναδική επαναλαμβάνουμε την διαδικασία



Αποφυγή Αδιεξόδου

- ▶ Καμία διεργασία δεν μπορεί να αποφασίσει μονομερώς να σταματήσει
- ▶ Πρέπει πρώτα να σιγουρευτεί ότι υπάρχει τουλάχιστον ένας ακόμα που προσπαθεί να εκλεγεί
- ▶ Σε αντίθετη περίπτωση
 - ▶ Έστω κατάσταση όπου καμία διεργασία δεν ενδιαφέρεται να εκλεγεί αρχηγός
 - ▶ Ο αλγόριθμος πέφτει σε αδιέξοδο
- ▶ Αντιμετωπίζουμε το πρόβλημα ως εξής:
 - ▶ Σε κάθε φάση απενεργοποιούνται όσες διεργασίες **καταλαβαίνουν** ότι δεν μπορούν να εκλεγούν
 - ▶ Έχουν δηλαδή μάθει ότι κάποια άλλη διεργασία είναι ισχυρότερη υποψήφια
 - ▶ Δεν είναι δυνατόν να παραιτηθούν όλες οι διεργασίες μαζί σε μία φάση



Ο Αλγόριθμος των Itai και Rodeh

Αλγόριθμος IR

Οι διεργασίες διατηρούν μια μεταβλητή **αρχηγός**=false και μια μεταβλητή **φάση**=0. Σε κάθε φάση οι διεργασίες διαλέγουν μια ταυτότητα από το σύνολο $\{1, \dots, n\}$ και την στέλνουν στον δεξιόστροφο γείτονα τους ως εξής: (**φάση, ταυτότητα, μετρίτης, μοναδική**). Ο **μετρίτης**=0 και **μοναδική**=true. Μόλις λάβουν μία ταυτότητα από τον αριστερόστροφο γείτονα, την συγκρίνουν με την δικιά τους. Αν είναι μεγαλύτερη, την προωθούν στον δεξιόστροφο γείτονα και αυξάνουν τον μετρίτη. Αν είναι μικρότερη, δεν κάνουν τίποτα. Αν είναι ίδια, ελέγχουν τον **μετρίτη**: αν είναι μικρότερος από n τότε θέτει την λογική μεταβλητή **μοναδική**=false και προωθούν το μήνυμα. Αν **μετρίτης** $\geq n$ και **μοναδική**=false τότε διαλέγουν μια νέα ταυτότητα, θέτουν **φάση++** και επαναλαμβάνουν τον αλγόριθμο. Αλλιώς μεταβαίνουν στην κατάσταση **εκλεγμένη** θέτοντας την μεταβλητή **αρχηγός** στην τιμή true.



Χαρακτηριστικά του Αλγόριθμου IR

- ▶ Ο αλγόριθμος εξελίσσεται σε φάσεις
 - ▶ Κάθε φάση διαρκεί $\mathcal{O}(n)$ γύρους
 - ▶ Οι διεργασίες ανταλλάσσουν $\mathcal{O}(n \log n)$ μηνύματα
 - ▶ Μπορούμε να δείξουμε ότι ο αριθμός των απαιτούμενων φάσεων για την εκλογή αρχηγού είναι $\frac{en}{n-1}$
 - ▶ Δηλαδή σταθερός αριθμός φάσεων $\mathcal{O}(1)$
- ▶ Η χρονική πολυπλοκότητα είναι $\mathcal{O}(n)$
- ▶ Η πολυπλοκότητα επικοινωνίας είναι $\mathcal{O}(n \log n)$



Πολυπλοκότητα Επικοινωνίας του Αλγόριθμου IR (1)

- ▶ Ας εξετάσουμε μια φάση του αλγορίθμου
- ▶ Έστω $p(i, d)$ η πιθανότητα το μήνυμα της διεργασίας που διάλεξε ταυτότητα i να ταξιδέψει απόσταση d

$$p(i, d) = \begin{cases} p_i^{n-1} & \text{if } d = n \\ p_i^{d-1}(1 - p_i) & \text{otherwise} \end{cases}$$

- ▶ Μας ενδιαφέρει να υπολογίσουμε την μέση απόσταση D που ταξιδεύει κάθε μήνυμα
 - ▶ Αυτό εξαρτάτε από την ταυτότητα i
 - ▶ Αν $i = n$ τότε $\bar{D} = n$
 - ▶ διαφορετικά αν $i < n$ τότε υπολογίζεται ως εξής



Πολυπλοκότητα Επικοινωνίας του Αλγόριθμου IR (2)

$$\begin{aligned} \bar{D}(i) &= \sum_{d=1}^n d \cdot p(i, d) \\ &= n \cdot p_i^{n-1} + (1 - p_i) \sum_{d=1}^{n-1} d \cdot p_i^{d-1} \\ &= \frac{1 - np_i^{n-1} + (n-1)p_i^n}{1 - p_i} + np_i^{n-1} \\ &= \frac{1 - p_i^n}{1 - p_i}, \quad 1 \leq i \leq n \end{aligned}$$

- ▶ Κάθε διεργασία έχει πιθανότητα $\frac{1}{n}$ να διαλέξει σαν ταυτότητα τον αριθμό i
- ▶ Η διεργασία με ταυτότητα i έχει πιθανότητα $\frac{i}{n}$ να εκλεχθεί αρχηγός



Πολυπλοκότητα Επικοινωνίας του Αλγόριθμου IR (3)

- ▶ Οπότε ο μέσος αριθμός μηνυμάτων N σε μία φάση φράσσεται από:

$$\begin{aligned} N &\leq \sum_{i=1}^n \bar{D}(i) = n + \sum_{i=1}^{n-1} \frac{1 - (\frac{i}{n})^n}{1 - \frac{i}{n}} \\ &\leq n + \sum_{i=1}^{n-1} \frac{1}{1 - \frac{i}{n}} = n + n \sum_{i=1}^{n-1} \frac{1}{n-i} \\ &= n + n \cdot \mathcal{O}(\log n) \end{aligned}$$

- ▶ Άρα ο αριθμός μηνυμάτων σε μία φάση είναι $\mathcal{O}(n \log n)$



Ο Αλγόριθμος FloodMax

Αλγόριθμος FloodMax

Οι διεργασίες διατηρούν μια μεταβλητή **αρχηγός** η οποία αρχικά είναι false και μια μεταβλητή **μέγιστη_ταυτότητα** με αρχική τιμή την ταυτότητα της διεργασίας. Σε κάθε γύρο, οι διεργασίες εκπέμπουν την **μέγιστη_ταυτότητα** σε όλους τους γείτονες. Μόλις λάβουν μία ταυτότητα απο κάποιον γείτονα, την συγκρίνουν με την **μέγιστη_ταυτότητα**. Αν είναι μεγαλύτερη, θέτουν την μεταβλητή στην νέα τιμή. Μετά απο δ γύρους, αν η μεταβλητή ισούται με την ταυτότητα της διεργασίας, η διεργασία μεταβαίνει στην κατάσταση **εκλεγμένη** θέτοντας την μεταβλητή **αρχηγός** στην τιμή true.

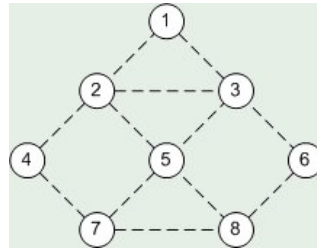
- ▶ Δεν γνωρίζουν το πλήθος των διεργασιών (n)
- ▶ Γνωρίζουν την διάμετρο του γραφήματος $\delta = \text{diam}(G)$
- ▶ Βασίζεται σε απλές πράξεις σύγκρισης ταυτοτήτων



Παράδειγμα Εκτέλεσης Αλγόριθμου FloodMax

- ▶ Έστω ένα σύγχρονο καταναμημένο σύστημα από $n = 8$ διεργασίες.
 - ▶ Γενικό δίκτυο $\delta = 3$
 - ▶ Οι διεργασίες είναι αριθμημένες από 1 έως 8

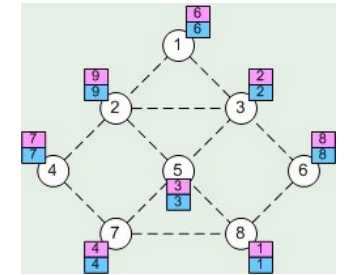
Διεργασίες σε δακτύλιο



Παράδειγμα Εκτέλεσης Αλγόριθμου FloodMax

- ▶ Έστω ένα σύγχρονο καταναμημένο σύστημα από $n = 8$ διεργασίες.
 - ▶ Γενικό δίκτυο $\delta = 3$
 - ▶ Οι διεργασίες είναι αριθμημένες από 1 έως 8
- ▶ Οι διεργασίες έχουν μοναδικές ταυτότητες
 - ▶ Δεν γνωρίζουν την ταυτότητα των υπόλοιπων διεργασιών

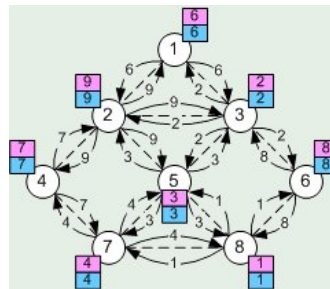
Διεργασίες σε δακτύλιο



Παράδειγμα Εκτέλεσης Αλγόριθμου FloodMax

- ▶ Έστω ένα σύγχρονο καταναμημένο σύστημα από $n = 8$ διεργασίες.
 - ▶ Γενικό δίκτυο $\delta = 3$
 - ▶ Οι διεργασίες είναι αριθμημένες από 1 έως 8
- ▶ Οι διεργασίες έχουν μοναδικές ταυτότητες
 - ▶ Δεν γνωρίζουν την ταυτότητα των υπόλοιπων διεργασιών
- ▶ Πρώτος Γύρος – αποστολή μηνυμάτων

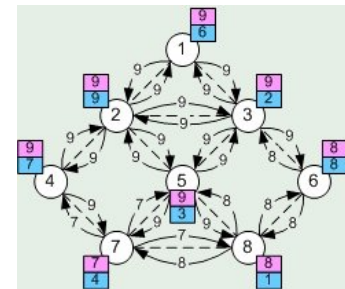
Διεργασίες σε δακτύλιο



Παράδειγμα Εκτέλεσης Αλγόριθμου FloodMax

- ▶ Έστω ένα σύγχρονο καταναμημένο σύστημα από $n = 8$ διεργασίες.
 - ▶ Γενικό δίκτυο $\delta = 3$
 - ▶ Οι διεργασίες είναι αριθμημένες από 1 έως 8
- ▶ Οι διεργασίες έχουν μοναδικές ταυτότητες
 - ▶ Δεν γνωρίζουν την ταυτότητα των υπόλοιπων διεργασιών
- ▶ Πρώτος Γύρος – αποστολή μηνυμάτων
- ▶ Δεύτερος Γύρος

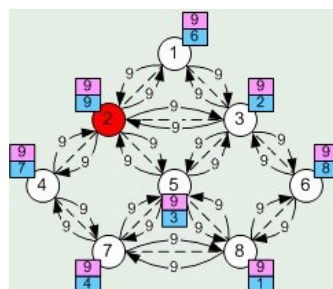
Διεργασίες σε δακτύλιο



Παράδειγμα Εκτέλεσης Αλγόριθμου FloodMax

- ▶ Έστω ένα σύγχρονο καταναμημένο σύστημα από $n = 8$ διεργασίες.
 - ▶ Γενικό δίκτυο $\delta = 3$
 - ▶ Οι διεργασίες είναι αριθμημένες από 1 έως 8
- ▶ Οι διεργασίες έχουν μοναδικές ταυτότητες
 - ▶ Δεν γνωρίζουν την ταυτότητα των υπόλοιπων διεργασιών
- ▶ Πρώτος Γύρος – αποστολή μηνυμάτων
- ▶ Δεύτερος Γύρος
- ▶ **Εκλογή αρχηγού – διεργασία 2**

Διεργασίες σε δακτύλιο



Χαρακτηριστικά του Αλγόριθμου FloodMax

Έστω n διεργασίες και m κανάλια, όπου η διεργασία με τη μεγαλύτερη ταυτότητα είναι η i_{max}

- ▶ Η διεργασία i_{max} εκλέγεται αρχηγός στο τέλος του γύρου δ
- ▶ Καμία διεργασία εκτός της i_{max} δεν είναι σε κατάσταση 'εκλεγμένη'
- ▶ Η χρονική πολυπλοκότητα είναι $\mathcal{O}(diam(G))$
- ▶ Η πολυπλοκότητα επικοινωνίας είναι $\mathcal{O}(diam(G) \cdot m)$

Απόδειξη Ορθότητας

Θεώρημα

Στον αλγόριθμο FloodMax η διεργασία i_{max} εκλέγεται αρχηγός στο τέλος του γύρου δ

Απόδειξη: Αρκεί να δείξουμε ότι μετά απο δ γύρους, η μεταβλητή **αρχηγός** _{i_{max}} = true.

Παρατηρούμε ότι μετά από r γύρους, η ταυτότητα της i_{max} έχει 'φτάσει' όλες τις διεργασίες που βρίσκονται σε απόσταση r από την i_{max} . Επομένως, στο τέλος του γύρου δ κάθε διεργασία θα έχει λάβει την ταυτότητα της i_{max} .

■

Σύνοψη 3ης Διάλεξης

Προηγούμενο Μάθημα

Προηγούμενο Μάθημα
Σύγχρονα Καταναμημένα Συστήματα
Εκλογή Αρχηγού

Πρόβλημα Εκλογής Αρχηγού

Καταναμημένοι Αλγόριθμοι σε Δίκτυα Δακτυλίου
Ανώνυμα Δίκτυα
Καταναμημένοι Αλγόριθμοι σε Γενικά Δίκτυα

Σύνοψη Μαθήματος

Σύνοψη Μαθήματος
Βιβλιογραφία
Επόμενο Μάθημα

Σύνοψη Μαθήματος

- ▶ Σύγχρονα Κατανεμημένα Συστήματα
- ▶ Πρόβλημα Εκλογής Αρχηγού
 - ▶ Δίκτυα Δακτυλίου
 - ▶ Αλγόριθμος των Hirschberg και Sinclair
 - ▶ Αλγόριθμος TimeSlice
 - ▶ Αλγόριθμος των Itai και Rodeh
 - ▶ Γενικά Δίκτυα
 - ▶ Αλγόριθμος FloodMax



Βιβλιογραφία

- ▶ Σημειώσεις του μαθήματος
- ▶ Βιβλίο "Distributed Algorithms" (N.Lynch)
 1. Κεφάλαιο 3: Leader Election in a Synchronous Ring
 2. Κεφάλαιο 4: Algorithms in General Synchronous Networks
- ▶ Βιβλίο "Distributed Computing Fundamentals, Simulations, and Advanced Topics" (H.Attyia, 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
 2. Κεφάλαιο 7: Election Algorithms
- ▶ Βιβλίο "Distributed Systems, Concepts and Design" (G.Coulouris, J.Dollimore, T.Kindberg)
 1. Κεφάλαιο 11: Coordination and Agreement



Επόμενο Μάθημα

- ▶ Σύγχρονα Κατανεμημένα Συστήματα
- ▶ Γενικά Δίκτυα
- ▶ Αναζήτηση Κατά Εύρος
- ▶ Συντομότερα Μονοπάτια

