

Κατανεμημένα Συστήματα I

Μάθημα Βασικής Επιλογής, Χειμερινού Εξαμήνου

Τομέας Εφαρμογών και Θεμελιώσεων

Ιωάννης Χατζηγιαννάκης

Δευτέρα, 2 Νοεμβρίου, 2009
Αίθουσα Β3



Προηγούμενο Μάθημα

- ▶ Σύγχρονα Κατανεμημένα Συστήματα
- ▶ Συναίνεση Υπό Την Παρουσία Σφαλμάτων
 - ▶ Σφάλματα Επικοινωνίας
 - ▶ Σφάλματα Τερματισμού
- ▶ Συντονισμός Στρατηγών
- ▶ Επικύρωση Δοσοληψιών
- ▶ Μελέτη Ορισμένων Κατανεμημένων Αλγόριθμων – Γενικά Δίκτυα



Το Πρόβλημα της Συναίνεσης

Υποθέτουμε ένα δίκτυο από n διεργασίες, συνδεδεμένες από ένα μη-κατευθυνόμενο γράφημα, όπου κάθε διεργασία γνωρίζει τη δομή του γραφήματος.

Κάθε διεργασία u δέχεται ως είσοδο μία τιμή i_u από το σύνολο S , δηλ.

$i_u \in S$

Ένας αλγόριθμος λύνει το πρόβλημα συναίνεσης εφόσον πληροί τις παρακάτω προδιαγραφές:

- Συμφωνία:** Κανένα ζεύγος διεργασιών δεν αποφασίζει διαφορετικές τιμές εξόδου, δηλ. $\nexists u, v : o_u \neq o_v$
- Εγκυρότητα:** Αν όλες οι διεργασίες αρχίζουν με την ίδια τιμή $i \in S$ δηλ. $\forall u \in [1, n] : i_u = i$, τότε η τιμή i είναι η μόνη πιθανή τελική απόφαση, δηλ. $\forall u \in [1, n] : o_u = i$
- Τερματισμός:** Όλες οι διαδικασίες τελικά αποφασίζουν



Παρουσία σφαλμάτων

Τι συμβαίνει όταν κατά την εκτέλεση ενός κατανεμημένου αλγορίθμου

1. παρουσιάζονται σφάλματα κατά την αποστολή μηνυμάτων
2. σφάλματα που παρουσιάζονται στις υπολογιστικές μονάδες (στους επεξεργαστές)

Υπό την παρουσία σφαλμάτων μπορούμε

- ▶ να εγγυηθούμε την ορθότητα του αλγορίθμου ;
- ▶ να εντοπίσουμε μια αποτυχία;
- ▶ να αντιμετωπίσουμε μια αποτυχία;



Μοντέλο Σφαλμάτων Επικοινωνίας

Εξετάζουμε την περίπτωση όπου

- ▶ κατά την εκτέλεση ενός κατανεμημένου αλγορίθμου
- ▶ παρουσιάζονται σφάλματα κατά την αποστολή μηνυμάτων

Σφάλμα Επικοινωνίας

Το δίκτυο επικοινωνίας που συνδέει τις μονάδες ενός κατανεμημένου συστήματος μπορεί να αποτύχει κατά την αποστολή ενός μηνύματος μέσω ενός (ελαττωματικού) καναλιού. Η παράδοση των μηνυμάτων που έχουν σταλεί δεν είναι εγγυημένη. Υποθέτουμε ότι ένας αριθμός μηνυμάτων που θα αποσταλούν κατά την εκτέλεση ενός κατανεμημένου αλγορίθμου, δεν θα παραδοθούν με επιτυχία.



Αδυναμία Εύρεσης Λύσης

Θεώρημα

Έστω σύγχρονο δίκτυο G που αποτελείται από δύο διεργασίες $V = \{u, v\}$ συνδεδεμένες από μια μη-κατευθυνόμενη ακμή uv . Δεν υπάρχει αλγόριθμος που να λύνει το πρόβλημα της συναίνεσης υπό την παρουσία σφαλμάτων επικοινωνίας.

- ▶ Βασικός περιορισμός στις δυνατότητες των κατανεμημένων δικτύων
- ▶ Για να ξεπεραστεί πρέπει να κάνουμε πιο ισχυρό το μοντέλο
 - ▶ Υποθέτουμε ότι ένας φραγμένος αριθμός μηνύματα κάνονται
 - ▶ Υποθέτουμε ότι τα μηνύματα κάνονται με πιθανότητα p
- ▶ Για να ξεπεραστεί πρέπει να κάνουμε πιο ασθενές το πρόβλημα
 - ▶ Επιτρέπουμε την παραβίαση της συνθήκης της συμφωνίας υπό ορισμένες συνθήκες
 - ▶ Επιτρέπουμε την παραβίαση της συνθήκης της εγκυρότητας υπό ορισμένες συνθήκες



Μοντέλο Σφαλμάτων Τερματισμού

Εξετάζουμε την περίπτωση όπου

- ▶ κατά την εκτέλεση ενός κατανεμημένου αλγορίθμου
- ▶ σφάλματα που παρουσιάζονται στις υπολογιστικές μονάδες (στους επεξεργαστές)
- ▶ εμφανίζονται το πολύ σ σφάλματα

Σφάλμα Τερματισμού

Κάποια (ελαττωματική) μονάδα του κατανεμημένου συστήματος μπορεί να αποτύχει κατά την εκτέλεση μια διεργασίας. Ένα σφάλμα τερματισμού μπορεί να παρουσιαστεί σε οποιοδήποτε σημείο της εκτέλεσης μιας διεργασίας. Η διεργασία μπορεί να τερματίσει ξαφνικά κατά την παραγωγή μηνυμάτων, οπότε να σταλεί μόνο ένα μέρος των εξερχόμενων μηνυμάτων.



Ο Αλγόριθμος FloodSet

Αλγόριθμος Συναίνεσης FloodSet

Κάθε διεργασία $u \in [1, n]$ διατηρεί μια λίστα l_u με **πμές εισόδου**, η οποία αρχικά περιέχει ένα μόνο την τιμή εισόδου $l_u \in S$ της u , $l_u = \{l_u\}$. Σε κάθε γύρο, οι διεργασίες εκπέμπουν την λίστα l σε όλους τους γείτονες. Μόλις λάβουν μια λίστα l_v από κάποιον γείτονα v , την ενοποιούν με την δικιά τους. Μετά από $\sigma + 1$ γύρους, αν η λίστα l_u περιέχει μία μόνο τιμή ($|l_u| = 1$) η διεργασία u αποφασίζει την τιμή που περιέχει η λίστα – αλλιώς αποφασίζει την προκαθορισμένη τιμή $p \in S$.

- ▶ Το γράφημα G είναι πλήρες
- ▶ Γνωρίζουν το συνολικό αριθμό σφαλμάτων σ
- ▶ Συμβολίζουμε με $l_u(\gamma)$ την τιμή της λίστας l_u που διατηρεί η διεργασία u τον γύρο γ



Πρόβλημα Επικύρωσης Δοσοληψιών

Οι διεργασίες του συστήματος, συμμετέχουν στην διεκπεραίωση μιας δοσοληψίας. Η κάθε διεργασία, σύμφωνα με τις τοπικές πληροφορίες εκφέρει μια άποψη για το αν η δοσοληψία πρέπει να επικυρωθεί – δηλαδή η διεκπεραίωση της ήταν επιτυχής. Στην συνέχεια οι διεργασίες επικοινωνούν μεταξύ τους για να πάρουν μια κοινή απόφαση – αν όλες συναινούν στην επικύρωση, τότε η δοσοληψία μπορεί να επικυρωθεί, αλλιώς όχι

- ▶ Οι πιθανές τιμές εισόδου/εξόδου μπορεί να είναι 'ναι' και 'όχι' – δηλ. $S = \{ "ναι", "όχι" \}$
- ▶ Αλγόριθμοι TwoPhaseCommit, ThreePhaseCommit



- ▶ Το δίκτυο περιέχει ελαττωματικές διεργασίες που δεν σταματούν αλλά συνεχίζουν να συμμετέχουν στην εκτέλεση του αλγορίθμου.
- ▶ Η συμπεριφορά των διεργασιών μπορεί να είναι τελείως ανεξέλεγκτη.
- ▶ Η εσωτερική κατάσταση μια ελαττωματικής διεργασίας μπορεί να αλλάξει κατά την διάρκεια ενός γύρου χωρίς να υπάρχει κάποιο μήνυμα.
- ▶ Μια ελαττωματική διεργασία μπορεί να στείλει μηνύματα με οποιοδήποτε περιεχόμενο, ανεξάρτητα από τις οδηγίες του καταμεμημένου αλγορίθμου που θα έπρεπε να τρέχει.
- ▶ Ονομάζουμε τέτοιου είδους σφάλματα ως **Βυζαντινά σφάλματα**.
- ▶ Μπορούμε να μοντελοποιήσουμε εχθρική συμπεριφορά (π.χ. θέματα ασφάλειας).



Συντονισμένη Επίθεση 4 Βυζαντινών Στρατηγών

Τέσσερις στρατηγοί διοικούν από ένα στρατό και θέλουν να εισβάλουν συντονισμένα σε μια αντίπαλη πόλη. Ανάμεσα στους στρατηγούς υπάρχει και ένας προδότης. Όλοι οι πιστοί στρατηγοί πρέπει να συμφωνήσουν στο ίδιο πλάνο επίθεσης (ή υποχώρησης) ανεξάρτητα από τις κινήσεις του προδότη. Οι επικοινωνία μεταξύ των αρχηγών γίνεται με ανταλλαγή μηνυμάτων. Ο προδότης έχει ελευθερία κινήσεων.

- ▶ Πρόβλημα συναίνεσης σε ένα σύστημα με $n = 4$ διεργασίες υπό την παρουσία βυζαντινών σφαλμάτων
- ▶ Οι πιθανές τιμές εισόδου/εξόδου μπορεί να είναι 'ναι' και 'όχι' – δηλ. $S = \{ "ναι", "όχι" \}$



- ▶ Ο ένας στρατηγός αναλαμβάνει την διοίκηση όλων των στρατών.
- ▶ Ο αρχιστράτηγος πρέπει να στείλει μια εντολή στους $n - 1$ στρατηγούς έτσι ώστε:
 1. Όλοι οι πιστοί στρατηγοί πρέπει να ακολουθήσουν την ίδια εντολή (όλες οι μη ελαττωματικές διεργασίες λαμβάνουν το ίδιο μήνυμα)
 2. Αν ο αρχιστράτηγος είναι πιστός, τότε όλοι οι πιστοί στρατηγοί ακολουθούν τις εντολές του (Αν όλες οι διεργασίες είναι μη ελαττωματικές τότε τα μηνύματα που λαμβάνονται είναι ίδια με αυτό που έστειλε η διεργασία συντονιστής)
- ▶ Οι παραπάνω συνθήκες είναι γνωστές ως συνθήκες **συνεπής εκπομπής**.
- ▶ Σημείωση: Αν ο αρχιστράτηγος είναι πιστός, η 1η συνθήκη προκύπτει από την 2η. Όμως ο αρχιστράτηγος μπορεί να είναι προδότης.



Συζήτηση Προβλήματος

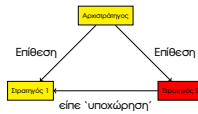
- ▶ Μια λύση στο πρόβλημα των Βυζαντινών Στρατηγών επιτρέπει:
 1. Αξίωση επικοινωνία υπό την παρουσία ψευδών μηνυμάτων
 2. Αξίωση επικοινωνία υπό την παρουσία παράληψης μηνυμάτων
- ▶ Η ανιμετώπιση σφαλμάτων παράληψης (σφάλματα τερματισμού/επικοινωνίας) είναι η ποιό συνηθισμένη προσέγγιση.
- ▶ Ονομάζουμε Βυζαντινά σφάλματα όλα τα σφάλματα που ανήκουν και στις δύο κατηγορίες.
- ▶ Όλες οι λύσεις στο πρόβλημα απαιτούν το πλήθος των διεργασιών να είναι **τουλάχιστον τριπλάσιο των σφαλμάτων** – δηλ. $n > 3\sigma$.
 - ▶ Είναι διαφορετικό από τα σφάλματα τερματισμού όπου το n και σ δεν είχαν κάποια σχέση.
 - ▶ Ίσως να ακούγεται υπερβολικό, καθότι γνωρίζουμε την τεχνική της *triple-modular redundancy* – δηλ. $n > 2\sigma + 1$.



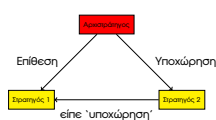
Αδυναμία Εύρεσης Λύσης (1)

Ας εξετάσουμε τις παρακάτω περιπτώσεις με 3 στρατηγούς:

Περίπτωση #1



Περίπτωση #2



- ▶ Στην 1η περίπτωση, ο Στρατηγός 1 για να ικανοποιήσει την δεύτερη συνθήκη θα πρέπει να επιτεθεί

2η Συνθήκη

Αν ο αρχιστράτηγος είναι πιστός, τότε όλοι οι πιστοί στρατηγοί ακολουθούν τις εντολές του



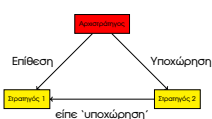
Αδυναμία Εύρεσης Λύσης (2)

Ας εξετάσουμε τις παρακάτω περιπτώσεις με 3 στρατηγούς:

Περίπτωση #1



Περίπτωση #2



- ▶ Στην 2η περίπτωση αν ο Στρατηγός 1 επιτεθεί τότε παραβιάζει την πρώτη συνθήκη

1η Συνθήκη

Όλοι οι πιστοί στρατηγοί πρέπει να ακολουθήσουν την ίδια εντολή



Αδυναμία Εύρεσης Λύσης (3)

- ▶ Με βάση τα μηνύματα που λαμβάνει ο Στρατηγός 1, σε κάθε περίπτωση οι δύο περιπτώσεις είναι ίδιες.
- ▶ Ο Στρατηγός 1 δεν μπορεί να διαφοροποιήσει τις δύο περιπτώσεις.
- ▶ Δεν υπάρχει καμία λύση για το πρόβλημα των βυζαντινών στρατηγών για την περίπτωση των 3 στρατηγών με 1 προδότη.
- ▶ Γενίκευση της αδυναμίας εύρεσης λύσης: Καμία λύση με λιγότερους από $3\beta + 1$ στρατηγούς δεν μπορεί να ανιμετωπίσει β προδότες



Αλγόριθμος Lamport, Shostak και Pease (1)

L. Lamport, R. Shostak, M. Pease: "The Byzantine Generals Problem", ACM Transactions on Programming Languages and Systems, 4(3): pp 382-401, 1982.

- ▶ Ο αλγόριθμος κάνει τρεις υποθέσεις ως προς την επικοινωνία:
 1. Όλα τα μηνύματα που στέλνονται παραδίδονται σωστά
 2. Ο παραλήπτης γνωρίζει ποιος είναι ο αποστολέας του μηνύματος
 3. Η απουσία ενός μηνύματος μπορεί να εντοπιστεί
- ▶ Η 1η και 2η υπόθεση δεν επιτρέπει στον προδότη να παρεμβάλλεται στην επικοινωνία άλλων στρατηγών
- ▶ Η 3η υπόθεση αποτρέπει έναν προδότη να καθυστερήσει την επίθεση απλά επειδή δεν στέλνει μηνύματα
- ▶ Σε δίκτυα υπολογιστών οι συνθήκες 1 και 2 υπονοούν ότι οι επεξεργαστές είναι συνδεδεμένοι απ' ευθείας και ότι ένα σφάλμα επικοινωνίας συνηθίζεται στα β σφάλματα
- ▶ Το μοντέλο των σύγχρονων ΚΣ ικανοποιεί και τις τρεις υποθέσεις



Αλγόριθμος Lamport, Shostak και Pease (2)

- ▶ Έστω n διεργασίες και β σφάλματα
- ▶ Οι διεργασίες έχουν μια προκαθορισμένη απόφαση o_{def} που χρησιμοποιούν όταν ο αρχιστράτηγος είναι προδότης (π.χ. υποχώρηση)
- ▶ Ορίζουμε μια συνάρτηση $majority(o_1, \dots, o_{n-1}) = o$ όταν η πλειοψηφία των αποφάσεων $o_u = o$

Αλγόριθμος UM(n, D) (για 0 προδότες)

1. Ο αρχιστράτηγος στέλνει την απόφαση o σε όλους τους στρατηγούς
2. Οι στρατηγοί αποφασίζουν ο ή αν δεν λάβουν κάποιο μήνυμα αποφασίζουν o_{def}



Αλγόριθμος Lamport, Shostak και Pease (3)

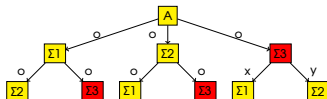
Αλγόριθμος UM(n, β) (για β προδότες)

1. Ο αρχιστράτηγος στέλνει την απόφαση o σε όλους τους στρατηγούς
2. Για κάθε στρατηγό u
 - ▶ Θέσε o_u την τιμή που έλαβε από τον αρχιστράτηγο ή αν δεν λάβουν κάποιο μήνυμα, θέσε o_{def}
 - ▶ Στείλε την τιμή o_u στους $n - 2$ στρατηγούς με τον UM($n-1, m-1$)
3. Για κάθε στρατηγό u και κάθε $v \neq u$
 - ▶ Θέσε o_v την τιμή που έλαβε από τον στρατηγό u που έλαβε στο βήμα 2 ή αν δεν έλαβε κάποιο μήνυμα, θέσε o_{def}
 - ▶ Αποφάσισε την τιμή $majority(o_1, \dots, o_{n-1})$



Παράδειγμα Εκτέλεσης (1)

$n = 4, \beta = 1$ - Ο Σ3 είναι προδότης

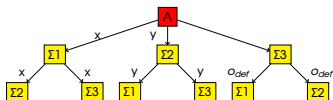


- ▶ Στο τέλος της 1ης φάσης: Σ1 ($o_1 = o$), Σ2 ($o_2 = o$), Σ3 ($o_3 = o$)
- ▶ Στο τέλος της 2ης φάσης:
 - Σ1 -- $o_1 = o, o_2 = o, o_3 = x$
 - Σ2 -- $o_1 = o, o_2 = o, o_3 = y$
 - Σ3 -- $o_1 = o, o_2 = o, o_3 = o$
- ▶ Στο τέλος της 2ης φάσης κάθε στρατηγός έχει λάβει ένα σύνολο τιμών και καταλήγει στο ίδιο συμπέρασμα λόγω της 1ης συνθήκης
- ▶ Η απόφαση του αρχιστράτηγου συμπίπτει με την πλειοψηφία (2η συνθήκη)



Παράδειγμα Εκτέλεσης (2)

$n = 4, \beta = 1$ – Ο A είναι προδότης



- ▶ Στο τέλος της 1ης φάσης: $\Sigma 1$ ($\alpha_1 = x$), $\Sigma 2$ ($\alpha_2 = y$), $\Sigma 3$ ($\alpha_3 = \alpha_{def}$)
- ▶ Στο τέλος της 2ης φάσης:
 - $\Sigma 1$ – $\alpha_1 = x, \alpha_2 = y, \alpha_3 = \alpha_{def}$
 - $\Sigma 2$ – $\alpha_1 = x, \alpha_2 = y, \alpha_3 = \alpha_{def}$
 - $\Sigma 3$ – $\alpha_1 = x, \alpha_2 = y, \alpha_3 = \alpha_{def}$
- ▶ Οι τρεις πιστοί στρατηγοί αποφασίζουν $\text{majority}(x, y, \alpha_{def})$ και οι συνθήκες 1 και 2 ικανοποιούνται



Ανάλυση Αλγόριθμου (1)

Λήμμα

Για κάθε m και k , ο $UM(m)$ ικανοποιεί της 2η συνθήκη εφόσον υπάρχουν $2k + m$ στρατηγοί και το πολύ k προδότες

Απόδειξη: (Με αναγωγή στο m)

Στο 1ο βήμα, ο $UM(0)$ λειτουργεί εφόσον ο αρχιστράτηγος είναι πιστός, δηλ. $UM(0)$ ικανοποιεί την 2η συνθήκη.

Ας υποθέσουμε ότι ο $UM(m-1)$ ικανοποιεί την 2η συνθήκη για $m > 0$.

Το αποδεικνύουμε για m :

- ▶ Στο 1ο βήμα ο πιστός στρατηγός στέλνει την τιμή α στους $n-1$ στρατηγούς.
- ▶ Στο 2ο βήμα όλοι οι πιστοί στρατηγοί εφαρμόζουν τον $UM(m-1)$.
- ▶ Από την υπόθεση ισχύει ότι $n > 2k + m$ ή $n-1 > 2k + (m-1)$.



Ανάλυση Αλγόριθμου (2)

- ▶ Από το βήμα αναγωγής που υποθέσαμε, κάθε πιστός στρατηγός u λαμβάνει $\alpha_u = \alpha_v$ από κάθε πιστό στρατηγό v .
- ▶ Εφόσον υπάρχουν το πολύ k προδότες και $n-1 > 2k + (m-1) \geq 2k$, δηλ. $k < \frac{n-1}{2}$ τότε η πλειοψηφία από τους $n-1$ είναι πιστοί.
- ▶ Επομένως κάθε πιστός στρατηγός έχει $\alpha_u = \alpha$ για την πλειοψηφία των $n-1$ τιμών άρα στο 3ο βήμα, εφαρμόζοντας $\text{majority}(\alpha_1, \dots, \alpha_{n-1})$ παίρνει ο u που ικανοποιεί την 2η συνθήκη.



Ανάλυση Αλγόριθμου (3)

Εξώρημα

Για κάθε m , ο $UM(m)$ ικανοποιεί την 1η και 2η συνθήκη εφόσον υπάρχουν $3m$ στρατηγοί και το πολύ m προδότες

Απόδειξη: (Με αναγωγή στο m)

Αν δεν υπάρχουν προδότες είναι εύκολο να δούμε ότι με την χρήση του αλγόριθμου, στο 1ο βήμα, οι συνθήκες 1 και 2 ικανοποιούνται.

Ας υποθέσουμε ότι ο $UM(m-1)$ ικανοποιεί τις συνθήκες 1 και 2 για $m > 0$. Το αποδεικνύουμε για m :

Περίπτωση 1:

- ▶ Υποθέτουμε ότι ο αρχιστράτηγος είναι πιστός.
- ▶ Όταν $k = m$ στο Λήμμα, τότε ο $UM(m)$ ικανοποιεί την 2η συνθήκη.
- ▶ Εφόσον η 1η συνθήκη προκύπτει από την 2η συνθήκη όταν ο αρχιστράτηγος είναι πιστός, αρκεί να ασχοληθούμε με την επόμενη περίπτωση:



Ανάλυση Αλγόριθμου (4)

Περίπτωση 2:

- ▶ Ο αρχιστράτηγος είναι προδότης.
- ▶ Υπάρχουν το πολύ m προδότες και ο αρχιστράτηγος είναι ένας από αυτούς.
- ▶ Άρα το πολύ $m - 1$ στρατηγοί είναι προδότες.
- ▶ Εφόσον υπάρχουν $3m$ στρατηγοί τότε οι πιστοί στρατηγοί πρέπει να είναι $3m - 1 > 3(m - 1)$
- ▶ Επομένως μπορούμε να εφαρμόσουμε το αναγκαστικό βήμα και να καταλήξουμε ότι ο $UM(m - 1)$ ικανοποιεί τις συνθήκες 1 και 2.
- ▶ Επομένως για κάθε v , κάθε ζευγάρι πιστών στρατηγών λαμβάνει την ίδια τιμή o_v στο 3ο βήμα.
- ▶ Επομένως κάθε ζευγάρι πιστών στρατηγών λαμβάνει το ίδιο σύνολο τιμών και άρα $\text{majority}(o_1, \dots, o_{m-1})$ δίνει την ίδια τιμή - που ικανοποιεί την 1η συνθήκη.



Ανάλυση Αλγόριθμου (5)

- ▶ Εφαρμόζοντας τον $UM(n, \beta)$ έχουμε $n - 1$ μηνύματα
- ▶ Για κάθε μήνυμα ενεργοποιείται ο $UM(n, \beta - 1)$ που στέλνει $n - 2$ μηνύματα
- ▶ ...
- ▶ Άρα ο συνολικός αριθμός μηνυμάτων είναι $O(n^{\beta+1})$
- ▶ Τα $\beta + 1$ βήματα όπου ανταλλάσσονται μηνύματα μεταξύ των διεργασιών είναι απαραίτητο χαρακτηριστικό των αλγορίθμων που πρέπει να καταλήξουν σε μια συναίνεση υπό την παρουσία β ελαττωματικών διεργασιών.



Σύνοψη 5^{ης} Διάλεξης

Προηγούμενο Μάθημα

Προηγούμενο Μάθημα
Βυζαντινά Σφάλματα

Ασύγχρονα Κατανεμημένα Συστήματα

Συζήτηση
Μοντελοποίηση Συστήματος
Βασικοί Κατανεμημένοι Αλγόριθμοι

Εύνοψη Μαθήματος

Εύνοψη Μαθήματος
Βιβλιογραφία
Επόμενο Μάθημα



1^η Ενότητα Μαθήματος

Εξετάσαμε τα Σύγχρονα Κατανεμημένα Συστήματα

1. Μοντελοποίηση συστημάτων
 - ▶ Διεργασίες, Δίκτυο Επικοινωνίας, Αλγόριθμοι, Σφάλματα, Πολυπλοκότητα
2. Θεμελιώδη προβλήματα
 - ▶ Εκλογή Αρχηγού, Αναζήτηση κατά Εύρος, Συντομότερα Μονοπάτια, Συναίνεση
 - ▶ Παραλλαγές
3. Χαρακτηριστικούς Κατανεμημένους Αλγόριθμους
 - ▶ LCR, FloodMax, BFS, FloodSet, 2/3-phase Commit
 - ▶ Παραλλαγές



Μελέτη Σύγχρονων Κατανεμημένων Συστημάτων (1)

- ▶ Η παραδοχή της συγχρονισμένης εκτέλεσης δεν αποτυπώνει πλήρως τις πραγματικές συνθήκες λειτουργίας
- ▶ Όμως, διευκολύνει την κατανόηση των προβλημάτων
 - ▶ Η παραδοχή της **συντονισμένης εκτέλεσης** των βημάτων των αλγόριθμων
 - ▶ Η υπόθεση **ταυτόχρονης παράδοσης** όλων των μηνυμάτων
- ▶ Εξετάσαμε την απόδοση των πρωτοκόλλων
 - ▶ ως προς τον χρόνο εκτέλεσης – **χρονική πολυπλοκότητα**
 - ▶ ως προς το πλήθος μηνυμάτων που ανταλλάχθηκαν – **πολυπλοκότητα επικοινωνίας**



Μελέτη Σύγχρονων Κατανεμημένων Συστημάτων (2)

- ▶ Μελετήσαμε την συμπεριφορά του συστήματος
 - ▶ παρουσία σφαλμάτων επικοινωνίας – τα μηνύματα μπορεί να χαθούν
 - ▶ παρουσία σφαλμάτων τερματισμού – οι διεργασίες μπορεί να αντιμετωπίσουν μόνιμα σφάλματα κατά την εκτέλεση
- ▶ Διερευνήσαμε τις δυνατότητες του συστήματος
 - ▶ αρνητικά αποτελέσματα – απόδειξη αδυναμίας αντιμετώπισης ενός προβλήματος
 1. Ανώνυμα δίκτυα – οι διεργασίες δεν έχουν μοναδικές ταυτότητες
 2. Παρουσία σφαλμάτων
- ▶ Ολοκληρώθηκε η πρώτη ενότητα του μαθήματος



Γενικά (1)

- ▶ Μία συλλογή υπολογιστικών μονάδων ή 'επεξεργαστές'
 - ▶ κάθε επεξεργαστής εκτελεί μόνο μία διεργασία
- ▶ Κάθε διεργασία u
 - ▶ 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 $msgs_u : states_u \times nbs_u^{out} \rightarrow M \cup \{\text{null}\}$
 - ▶ Διαθέτει μία συνάρτηση αλλαγής κατάστασης $trans_u : states_u \times (M \cup \{\text{null}\})^{nbs_u^i} \rightarrow states_u$



Γενικά (2)

Στο μοντέλο σύγχρονων κατανεμημένων συστημάτων, θεωρούμε ότι 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**



Εκτέλεση σε Σύγχρονο Κατανομημένο Σύστημα

- ▶ Παράδειγμα εκτέλεσης ενός Σύγχρονου Συστήματος
- ▶ Αρχικά
 - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
 - ▶ όλα τα κανάλια είναι άδεια
- ▶ οι διεργασίες, εκτελούν 'αυτονομιά' το πρωτόκολλο

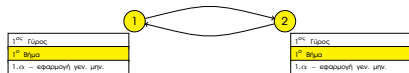
Εκτέλεση Σύγχρονου Συστήματος



Εκτέλεση σε Σύγχρονο Κατανομημένο Σύστημα

- ▶ Παράδειγμα εκτέλεσης ενός Σύγχρονου Συστήματος
- ▶ Αρχικά
 - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
 - ▶ όλα τα κανάλια είναι άδεια
- ▶ οι διεργασίες, εκτελούν 'αυτονομιά' το πρωτόκολλο

Εκτέλεση Σύγχρονου Συστήματος



Εκτέλεση σε Σύγχρονο Κατανομημένο Σύστημα

- ▶ Παράδειγμα εκτέλεσης ενός Σύγχρονου Συστήματος
- ▶ Αρχικά
 - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
 - ▶ όλα τα κανάλια είναι άδεια
- ▶ οι διεργασίες, εκτελούν 'αυτονομιά' το πρωτόκολλο

Εκτέλεση Σύγχρονου Συστήματος



Εκτέλεση σε Σύγχρονο Κατανομημένο Σύστημα

- ▶ Παράδειγμα εκτέλεσης ενός Σύγχρονου Συστήματος
- ▶ Αρχικά
 - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
 - ▶ όλα τα κανάλια είναι άδεια
- ▶ οι διεργασίες, εκτελούν 'αυτονομιά' το πρωτόκολλο

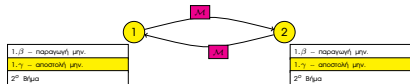
Εκτέλεση Σύγχρονου Συστήματος



Εκτέλεση σε Σύγχρονο Κατανομημένο Σύστημα

- ▶ Παράδειγμα εκτέλεσης ενός Σύγχρονου Συστήματος
- ▶ Αρχικά
 - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
 - ▶ όλα τα κανάλια είναι άδεια
- ▶ οι διεργασίες, εκτελούν 'συντονισμένα' το πρωτόκολλο

Εκτέλεση Σύγχρονου Συστήματος



Εκτέλεση σε Σύγχρονο Κατανομημένο Σύστημα

- ▶ Παράδειγμα εκτέλεσης ενός Σύγχρονου Συστήματος
- ▶ Αρχικά
 - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
 - ▶ όλα τα κανάλια είναι άδεια
- ▶ οι διεργασίες, εκτελούν 'συντονισμένα' το πρωτόκολλο

Εκτέλεση Σύγχρονου Συστήματος



Εκτέλεση σε Σύγχρονο Κατανομημένο Σύστημα

- ▶ Παράδειγμα εκτέλεσης ενός Σύγχρονου Συστήματος
- ▶ Αρχικά
 - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
 - ▶ όλα τα κανάλια είναι άδεια
- ▶ οι διεργασίες, εκτελούν 'συντονισμένα' το πρωτόκολλο

Εκτέλεση Σύγχρονου Συστήματος



Εκτέλεση σε Σύγχρονο Κατανομημένο Σύστημα

- ▶ Παράδειγμα εκτέλεσης ενός Σύγχρονου Συστήματος
- ▶ Αρχικά
 - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
 - ▶ όλα τα κανάλια είναι άδεια
- ▶ οι διεργασίες, εκτελούν 'συντονισμένα' το πρωτόκολλο

Εκτέλεση Σύγχρονου Συστήματος



Εκτέλεση σε Σύγχρονο Κατανεμημένο Σύστημα

- ▶ Παράδειγμα εκτέλεσης ενός Σύγχρονου Συστήματος
- ▶ Αρχικά
 - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
 - ▶ όλα τα κανάλια είναι άδεια
- ▶ οι διεργασίες, εκτελούν 'αυτονοσιμένα' το πρωτόκολλο

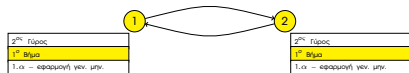
Εκτέλεση Σύγχρονου Συστήματος



Εκτέλεση σε Σύγχρονο Κατανεμημένο Σύστημα

- ▶ Παράδειγμα εκτέλεσης ενός Σύγχρονου Συστήματος
- ▶ Αρχικά
 - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
 - ▶ όλα τα κανάλια είναι άδεια
- ▶ οι διεργασίες, εκτελούν 'αυτονοσιμένα' το πρωτόκολλο

Εκτέλεση Σύγχρονου Συστήματος



Εκτέλεση σε Σύγχρονο Κατανεμημένο Σύστημα

- ▶ Παράδειγμα εκτέλεσης ενός Σύγχρονου Συστήματος
- ▶ Αρχικά
 - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
 - ▶ όλα τα κανάλια είναι άδεια
- ▶ οι διεργασίες, εκτελούν 'αυτονοσιμένα' το πρωτόκολλο

Εκτέλεση Σύγχρονου Συστήματος



Εκτέλεση σε Σύγχρονο Κατανεμημένο Σύστημα

- ▶ Παράδειγμα εκτέλεσης ενός Σύγχρονου Συστήματος
- ▶ Αρχικά
 - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
 - ▶ όλα τα κανάλια είναι άδεια
- ▶ οι διεργασίες, εκτελούν 'αυτονοσιμένα' το πρωτόκολλο

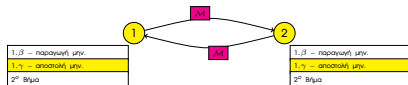
Εκτέλεση Σύγχρονου Συστήματος



Εκτέλεση σε Σύγχρονο Κατανομημένο Σύστημα

- ▶ Παράδειγμα εκτέλεσης ενός Σύγχρονου Συστήματος
- ▶ Αρχικά
 - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
 - ▶ όλα τα κανάλια είναι άδεια
- ▶ οι διεργασίες, εκτελούν 'συντονισμένα' το πρωτόκολλο

Εκτέλεση Σύγχρονου Συστήματος



Εκτέλεση σε Σύγχρονο Κατανομημένο Σύστημα

- ▶ Παράδειγμα εκτέλεσης ενός Σύγχρονου Συστήματος
- ▶ Αρχικά
 - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
 - ▶ όλα τα κανάλια είναι άδεια
- ▶ οι διεργασίες, εκτελούν 'συντονισμένα' το πρωτόκολλο

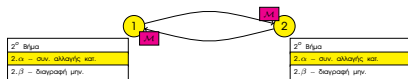
Εκτέλεση Σύγχρονου Συστήματος



Εκτέλεση σε Σύγχρονο Κατανομημένο Σύστημα

- ▶ Παράδειγμα εκτέλεσης ενός Σύγχρονου Συστήματος
- ▶ Αρχικά
 - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
 - ▶ όλα τα κανάλια είναι άδεια
- ▶ οι διεργασίες, εκτελούν 'συντονισμένα' το πρωτόκολλο

Εκτέλεση Σύγχρονου Συστήματος



Εκτέλεση σε Σύγχρονο Κατανομημένο Σύστημα

- ▶ Παράδειγμα εκτέλεσης ενός Σύγχρονου Συστήματος
- ▶ Αρχικά
 - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
 - ▶ όλα τα κανάλια είναι άδεια
- ▶ οι διεργασίες, εκτελούν 'συντονισμένα' το πρωτόκολλο

Εκτέλεση Σύγχρονου Συστήματος



Εκτέλεση σε Σύγχρονο Καταναμημένο Σύστημα

- ▶ Παράδειγμα εκτέλεσης ενός Σύγχρονου Συστήματος
- ▶ Αρχικά
 - ▶ όλες οι διεργασίες βρίσκονται σε κάποια αρχική κατάσταση
 - ▶ όλα τα κανάλια είναι άδεια
- ▶ οι διεργασίες, εκτελούν 'συντονισμένα' το πρωτόκολλο

Εκτέλεση Σύγχρονου Συστήματος



Επεξεργαστές Διαφορετικών ταχυτήτων (1)

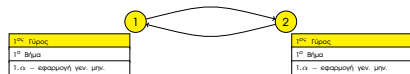
Το σύγχρονο μοντέλο υποθέτει ότι οι επεξεργαστές έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως οι διεργασίες, εκτελούν 'συντονισμένα' το πρωτόκολλο

Τι θα συμβεί αν οι επεξεργαστές δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες διεργασίες εκτελούν τα βήματα πιο αργά

Εκτέλεση με Επεξεργαστές διαφορετικών ταχυτήτων



Επεξεργαστές Διαφορετικών ταχυτήτων (1)

Το σύγχρονο μοντέλο υποθέτει ότι οι επεξεργαστές έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως οι διεργασίες, εκτελούν 'συντονισμένα' το πρωτόκολλο

Τι θα συμβεί αν οι επεξεργαστές δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες διεργασίες εκτελούν τα βήματα πιο αργά

Εκτέλεση με Επεξεργαστές διαφορετικών ταχυτήτων



Επεξεργαστές Διαφορετικών ταχυτήτων (1)

Το σύγχρονο μοντέλο υποθέτει ότι οι επεξεργαστές έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως οι διεργασίες, εκτελούν 'συντονισμένα' το πρωτόκολλο

Τι θα συμβεί αν οι επεξεργαστές δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες διεργασίες εκτελούν τα βήματα πιο αργά

Εκτέλεση με Επεξεργαστές διαφορετικών ταχυτήτων



Επεξεργαστές Διαφορετικών ταχυτήτων (1)

Το σύγχρονο μοντέλο υποθέτει ότι οι επεξεργαστές έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως οι διεργασίες, εκτελούν 'αυτονοσιμμένα' το πρωτόκολλο

Τι θα συμβεί αν οι επεξεργαστές δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες διεργασίες εκτελούν τα βήματα πιο αργά

Εκτέλεση με Επεξεργαστές διαφορετικών ταχυτήτων



Επεξεργαστές Διαφορετικών ταχυτήτων (1)

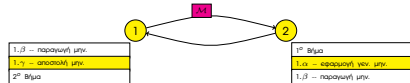
Το σύγχρονο μοντέλο υποθέτει ότι οι επεξεργαστές έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως οι διεργασίες, εκτελούν 'αυτονοσιμμένα' το πρωτόκολλο

Τι θα συμβεί αν οι επεξεργαστές δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες διεργασίες εκτελούν τα βήματα πιο αργά

Εκτέλεση με Επεξεργαστές διαφορετικών ταχυτήτων



Επεξεργαστές Διαφορετικών ταχυτήτων (1)

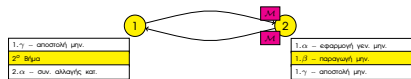
Το σύγχρονο μοντέλο υποθέτει ότι οι επεξεργαστές έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως οι διεργασίες, εκτελούν 'αυτονοσιμμένα' το πρωτόκολλο

Τι θα συμβεί αν οι επεξεργαστές δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες διεργασίες εκτελούν τα βήματα πιο αργά

Εκτέλεση με Επεξεργαστές διαφορετικών ταχυτήτων



Επεξεργαστές Διαφορετικών ταχυτήτων (1)

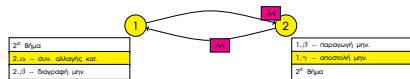
Το σύγχρονο μοντέλο υποθέτει ότι οι επεξεργαστές έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως οι διεργασίες, εκτελούν 'αυτονοσιμμένα' το πρωτόκολλο

Τι θα συμβεί αν οι επεξεργαστές δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες διεργασίες εκτελούν τα βήματα πιο αργά

Εκτέλεση με Επεξεργαστές διαφορετικών ταχυτήτων



Επεξεργαστές Διαφορετικών ταχυτήτων (1)

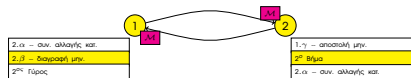
Το σύγχρονο μοντέλο υποθέτει ότι οι επεξεργαστές έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως οι διεργασίες, εκτελούν **'συντονισμένα'** το πρωτόκολλο

Τι θα συμβεί αν οι επεξεργαστές δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες διεργασίες εκτελούν τα βήματα πιο αργά

Εκτέλεση με Επεξεργαστές διαφορετικών ταχυτήτων



Επεξεργαστές Διαφορετικών ταχυτήτων (1)

Το σύγχρονο μοντέλο υποθέτει ότι οι επεξεργαστές έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως οι διεργασίες, εκτελούν **'συντονισμένα'** το πρωτόκολλο

Τι θα συμβεί αν οι επεξεργαστές δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες διεργασίες εκτελούν τα βήματα πιο αργά

Εκτέλεση με Επεξεργαστές διαφορετικών ταχυτήτων



Επεξεργαστές Διαφορετικών ταχυτήτων (1)

Το σύγχρονο μοντέλο υποθέτει ότι οι επεξεργαστές έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως οι διεργασίες, εκτελούν **'συντονισμένα'** το πρωτόκολλο

Τι θα συμβεί αν οι επεξεργαστές δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες διεργασίες εκτελούν τα βήματα πιο αργά

Εκτέλεση με Επεξεργαστές διαφορετικών ταχυτήτων



Επεξεργαστές Διαφορετικών ταχυτήτων (1)

Το σύγχρονο μοντέλο υποθέτει ότι οι επεξεργαστές έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως οι διεργασίες, εκτελούν **'συντονισμένα'** το πρωτόκολλο

Τι θα συμβεί αν οι επεξεργαστές δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες διεργασίες εκτελούν τα βήματα πιο αργά

Εκτέλεση με Επεξεργαστές διαφορετικών ταχυτήτων



Επεξεργαστές Διαφορετικών ταχυτήτων (1)

Το σύγχρονο μοντέλο υποθέτει ότι οι επεξεργαστές έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως οι διεργασίες, εκτελούν **'αυτονοσιμένα'** το πρωτόκολλο

Τι θα συμβεί αν οι επεξεργαστές δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες διεργασίες εκτελούν τα βήματα πιο αργά

Εκτέλεση με Επεξεργαστές διαφορετικών ταχυτήτων



Επεξεργαστές Διαφορετικών ταχυτήτων (1)

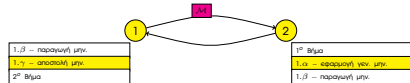
Το σύγχρονο μοντέλο υποθέτει ότι οι επεξεργαστές έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως οι διεργασίες, εκτελούν **'αυτονοσιμένα'** το πρωτόκολλο

Τι θα συμβεί αν οι επεξεργαστές δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες διεργασίες εκτελούν τα βήματα πιο αργά

Εκτέλεση με Επεξεργαστές διαφορετικών ταχυτήτων



Επεξεργαστές Διαφορετικών ταχυτήτων (1)

Το σύγχρονο μοντέλο υποθέτει ότι οι επεξεργαστές έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως οι διεργασίες, εκτελούν **'αυτονοσιμένα'** το πρωτόκολλο

Τι θα συμβεί αν οι επεξεργαστές δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες διεργασίες εκτελούν τα βήματα πιο αργά

Εκτέλεση με Επεξεργαστές διαφορετικών ταχυτήτων



Επεξεργαστές Διαφορετικών ταχυτήτων (1)

Το σύγχρονο μοντέλο υποθέτει ότι οι επεξεργαστές έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως οι διεργασίες, εκτελούν **'αυτονοσιμένα'** το πρωτόκολλο

Τι θα συμβεί αν οι επεξεργαστές δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες διεργασίες εκτελούν τα βήματα πιο αργά

Εκτέλεση με Επεξεργαστές διαφορετικών ταχυτήτων



Επεξεργαστές Διαφορετικών ταχυτήτων (1)

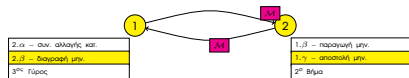
Το σύγχρονο μοντέλο υποθέτει ότι οι επεξεργαστές έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως οι διεργασίες, εκτελούν **‘αυτονομιμένα’** το πρωτόκολλο

Τι θα συμβεί αν οι επεξεργαστές δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες διεργασίες εκτελούν τα βήματα πιο αργά

Εκτέλεση με Επεξεργαστές διαφορετικών ταχυτήτων



Επεξεργαστές Διαφορετικών ταχυτήτων (1)

Το σύγχρονο μοντέλο υποθέτει ότι οι επεξεργαστές έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως οι διεργασίες, εκτελούν **‘αυτονομιμένα’** το πρωτόκολλο

Τι θα συμβεί αν οι επεξεργαστές δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες διεργασίες εκτελούν τα βήματα πιο αργά

Εκτέλεση με Επεξεργαστές διαφορετικών ταχυτήτων



Επεξεργαστές Διαφορετικών ταχυτήτων (2)

- ▶ Η εκτέλεση της διεργασίας 2 εξελίσσεται πιο αργά
- ▶ Τι γίνεται με τα μηνύματα που παραδίδονται ;
 - ▶ Υποθέτουμε κάποια ουρά εισερχόμενων μηνυμάτων ;
 - ▶ Πόσα μηνύματα μπορεί να αποθηκεύσει ;
 - ▶ Σε περίπτωση μικρής ουράς, χάνονται μηνύματα ;
- ▶ Η διεργασία 1 διαγράφει κάποια μηνύματα προτού τα επεξεργαστεί
- ▶ Τι άλλα προβλήματα μπορεί να προκύψουν σε ένα τέτοιο σύστημα ;

Κανάλια Επικοινωνίας Διαφορετικών ταχυτήτων (1)

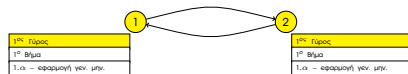
Το σύγχρονο μοντέλο υποθέτει ότι τα κανάλια επικοινωνίας έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως τα μηνύματα παραδίδονται **‘ταυτόχρονα’**

Τι θα συμβεί αν τα κανάλια επικοινωνίας δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες μηνύματα παραδίδονται αργότερα

Εκτέλεση με Κανάλια διαφορετικών ταχυτήτων



Κανάλια Επικοινωνίας Διαφορετικών ταχυτήτων (1)

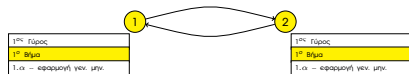
Το σύγχρονο μοντέλο υποθέτει ότι τα κανάλια επικοινωνίας έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως τα μηνύματα παραδίδονται **'ταυτόχρονα'**

Τι θα συμβεί αν τα κανάλια επικοινωνίας δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες μηνύματα παραδίδονται αργότερα

Εκτέλεση με Κανάλια διαφορετικών ταχυτήτων



Κανάλια Επικοινωνίας Διαφορετικών ταχυτήτων (1)

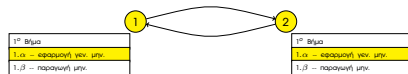
Το σύγχρονο μοντέλο υποθέτει ότι τα κανάλια επικοινωνίας έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως τα μηνύματα παραδίδονται **'ταυτόχρονα'**

Τι θα συμβεί αν τα κανάλια επικοινωνίας δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες μηνύματα παραδίδονται αργότερα

Εκτέλεση με Κανάλια διαφορετικών ταχυτήτων



Κανάλια Επικοινωνίας Διαφορετικών ταχυτήτων (1)

Το σύγχρονο μοντέλο υποθέτει ότι τα κανάλια επικοινωνίας έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως τα μηνύματα παραδίδονται **'ταυτόχρονα'**

Τι θα συμβεί αν τα κανάλια επικοινωνίας δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες μηνύματα παραδίδονται αργότερα

Εκτέλεση με Κανάλια διαφορετικών ταχυτήτων



Κανάλια Επικοινωνίας Διαφορετικών ταχυτήτων (1)

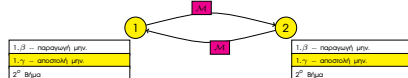
Το σύγχρονο μοντέλο υποθέτει ότι τα κανάλια επικοινωνίας έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως τα μηνύματα παραδίδονται **'ταυτόχρονα'**

Τι θα συμβεί αν τα κανάλια επικοινωνίας δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες μηνύματα παραδίδονται αργότερα

Εκτέλεση με Κανάλια διαφορετικών ταχυτήτων



Κανάλια Επικοινωνίας Διαφορετικών ταχυτήτων (1)

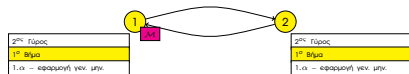
Το σύγχρονο μοντέλο υποθέτει ότι τα κανάλια επικοινωνίας έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως τα μηνύματα παραδίδονται **'ταυτόχρονα'**

Τι θα συμβεί αν τα κανάλια επικοινωνίας δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες μηνύματα παραδίδονται αργότερα

Εκτέλεση με Κανάλια διαφορετικών ταχυτήτων



Κανάλια Επικοινωνίας Διαφορετικών ταχυτήτων (1)

Το σύγχρονο μοντέλο υποθέτει ότι τα κανάλια επικοινωνίας έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως τα μηνύματα παραδίδονται **'ταυτόχρονα'**

Τι θα συμβεί αν τα κανάλια επικοινωνίας δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες μηνύματα παραδίδονται αργότερα

Εκτέλεση με Κανάλια διαφορετικών ταχυτήτων



Κανάλια Επικοινωνίας Διαφορετικών ταχυτήτων (1)

Το σύγχρονο μοντέλο υποθέτει ότι τα κανάλια επικοινωνίας έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως τα μηνύματα παραδίδονται **'ταυτόχρονα'**

Τι θα συμβεί αν τα κανάλια επικοινωνίας δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες μηνύματα παραδίδονται αργότερα

Εκτέλεση με Κανάλια διαφορετικών ταχυτήτων



Κανάλια Επικοινωνίας Διαφορετικών ταχυτήτων (1)

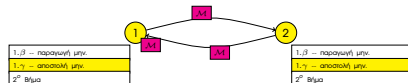
Το σύγχρονο μοντέλο υποθέτει ότι τα κανάλια επικοινωνίας έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως τα μηνύματα παραδίδονται **'ταυτόχρονα'**

Τι θα συμβεί αν τα κανάλια επικοινωνίας δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες μηνύματα παραδίδονται αργότερα

Εκτέλεση με Κανάλια διαφορετικών ταχυτήτων



Κανάλια Επικοινωνίας Διαφορετικών ταχυτήτων (1)

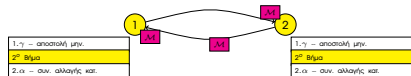
Το σύγχρονο μοντέλο υποθέτει ότι τα κανάλια επικοινωνίας έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως τα μηνύματα παραδίδονται **'ταυτόχρονα'**

Τι θα συμβεί αν τα κανάλια επικοινωνίας δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες μηνύματα παραδίδονται αργότερα

Εκτέλεση με Κανάλια διαφορετικών ταχυτήτων



Κανάλια Επικοινωνίας Διαφορετικών ταχυτήτων (1)

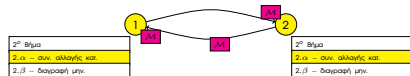
Το σύγχρονο μοντέλο υποθέτει ότι τα κανάλια επικοινωνίας έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως τα μηνύματα παραδίδονται **'ταυτόχρονα'**

Τι θα συμβεί αν τα κανάλια επικοινωνίας δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες μηνύματα παραδίδονται αργότερα

Εκτέλεση με Κανάλια διαφορετικών ταχυτήτων



Κανάλια Επικοινωνίας Διαφορετικών ταχυτήτων (1)

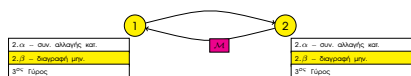
Το σύγχρονο μοντέλο υποθέτει ότι τα κανάλια επικοινωνίας έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως τα μηνύματα παραδίδονται **'ταυτόχρονα'**

Τι θα συμβεί αν τα κανάλια επικοινωνίας δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες μηνύματα παραδίδονται αργότερα

Εκτέλεση με Κανάλια διαφορετικών ταχυτήτων



Κανάλια Επικοινωνίας Διαφορετικών ταχυτήτων (1)

Το σύγχρονο μοντέλο υποθέτει ότι τα κανάλια επικοινωνίας έχουν ίδια χαρακτηριστικά (ταχύτητα)

- ▶ επομένως τα μηνύματα παραδίδονται **'ταυτόχρονα'**

Τι θα συμβεί αν τα κανάλια επικοινωνίας δεν έχουν ίδιες ταχύτητες;

- ▶ Κάποιες μηνύματα παραδίδονται αργότερα

Εκτέλεση με Κανάλια διαφορετικών ταχυτήτων



Κανάλια Επικοινωνίας Διαφορετικών ταχυτήτων (2)

- ▶ Το κανάλι επικοινωνίας που συνδέει την διεργασία 2 με την 1 είναι πιο αργό απο το κανάλι επικοινωνίας που συνδέει την 1 με την 2
- ▶ Τι γίνεται όταν το κανάλι επικοινωνίας μεταδίδει ένα μήνυμα και η διεργασία επικειρήσει να στείλει ένα άλλο ;
 - ▶ Υποθέτουμε κάποια ουρά εξερχόμενων μηνυμάτων ;
 - ▶ Πόσα μηνύματα μπορεί να αποθηκεύσει ;
 - ▶ Σε περίπτωση μικρής ουράς, χάνονται μηνύματα ;
- ▶ Τι άλλα προβλήματα μπορεί να προκύψουν σε ένα τέτοιο σύστημα ;



Μοντελοποίηση Ασύγχρονων Συστημάτων (1)

Μελετάμε καταναμημένα συστήματα, όπου

1. τα επιμέρους συστήματα εκτελούν τις εργασίες τους
 - ▶ με **οποιαδήποτε, ακαθόριστη**, σειρά
 - ▶ με **οποιαδήποτε, ακαθόριστη, ταχύτητα** σε σχέση με τα υπόλοιπα υποσυστήματα
 - ▶ δεν κάνουμε καμία παραδοχή ως προς τον ρυθμό που εκτελεί εντολές
2. τα κανάλια επικοινωνίας παραδίδουν τα μηνύματα με **οποιαδήποτε, ακαθόριστη, ταχύτητα** σε σχέση με τα υπόλοιπα κανάλια
 - ▶ δεν κάνουμε καμία παραδοχή ως προς τον ρυθμό παράδοσης μηνυμάτων



Μοντελοποίηση Ασύγχρονων Συστημάτων (2)

- ▶ Μοντελοποιούμε αυτή την **απροσδιόριστη χρονικά συμπεριφορά** χρησιμοποιώντας **αυτόματα εισόδου/εξόδου**
 - ▶ Κάθε διεργασία μοντελοποιείτε απο ένα αυτόματο εισόδου/εξόδου
 - ▶ Κάθε κανάλι επικοινωνίας μοντελοποιείτε απο ένα αυτόματο εισόδου/εξόδου
- ▶ Το μοντέλο των αυτόματων εισόδου/εξόδου είναι αρκετά γενικό
 - ▶ Μπορούμε να το χρησιμοποιήσουμε για να περιγράψουμε σχεδόν όλους τους τύπους ασύγχρονων συστημάτων



Αυτόματα Εισόδου/Εξόδου (1)

- ▶ Ένα αυτόματο εισόδου/εξόδου μοντελοποιεί ένα στοιχείο του καταναμημένου συστήματος το οποίο αλληλεπιδρά με άλλα στοιχεία του συστήματος
- ▶ Είναι ένα **αυτόματο καταστάσεων** (state machine) όπου οι μεταβάσεις μεταξύ καταστάσεων συνδέονται με ένα σύνολο **ενεργειών**
- ▶ Οι **ενέργειες** κατηγοριοποιούνται ως εξής:
 1. Ενέργειες εισόδου
 2. Ενέργειες εξόδου
 3. Εσωτερικές ενέργειες



Αυτόματα Εισόδου/Εξόδου (2)

- ▶ Οι **ενέργειες εισόδου και εξόδου** χρησιμοποιούνται για την επικοινωνία με το περιβάλλον του αυτόματου
 - ▶ Παράδειγμα – μια ενέργεια εισόδου είναι η παραλαβή ενός μηνύματος απο μια γειτονική διεργασία
 - ▶ Παράδειγμα – μια ενέργεια εξόδου είναι η αποστολή ενός μηνύματος προς κάποιο κανάλι επικοινωνίας
- ▶ Οι **εσωτερικές ενέργειες** είναι ορατές μόνο στο αυτόματο
- ▶ Ένα αυτόματο δεν καθορίζει πότε και ποια ενέργεια εισόδου θα συμβεί – εξαρτάτε απο τα γειτονικά αυτόματα
 - ▶ Μπορεί να καθορίσει τις ενέργειες εξόδου και τις εσωτερικές ενέργειες

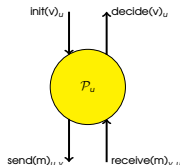


Παράδειγμα Αυτόματου Εισόδου/Εξόδου (1)

Παράδειγμα διεργασίας που συμμετέχει στην εκτέλεση ενός κατανημμένου αλγόριθμου συνείσεσης

- ▶ Η κατάσταση της P_u καθορίζεται από
 - ▶ τις τιμές των άλλων διεργασιών
 - ▶ αν έχει ανακοινώσει την απόφαση της
- ▶ Οι ενέργειες εισόδου είναι της μορφής $init(v)_u$ και $receive(m)_{u,v}$
- ▶ Οι ενέργειες εξόδου είναι της μορφής $decide(v)_u$ και $send(m)_{u,v}$

Η διεργασία P_u



Παράδειγμα Αυτόματου Εισόδου/Εξόδου (2)

Παράδειγμα καναλιού επικοινωνίας που συνδέει τις διεργασίες u , v και παραδίδει τα μηνύματα με τη σειρά που τα παρέλαβε (FIFO)

- ▶ Έστω \mathcal{M} το αλφάβητο μηνυμάτων
- ▶ Οι ενέργειες εισόδου είναι οι αποστολές μηνυμάτων απο την u προς την v
- ▶ Οι ενέργειες εξόδου είναι οι παραδόσεις μηνυμάτων στην v

Το κανάλι επικοινωνίας $C_{u,v}$



Ορισμός Αυτόματου Εισόδου/Εξόδου (1)

Ένα αυτόματο εισόδου/εξόδου \mathcal{A} αποτελείται από:

- ▶ Τρία σύνολα απο ενέργειες εισόδου $in(\mathcal{A})$ – ενέργειες εξόδου $out(\mathcal{A})$, και εσωτερικές ενέργειες $int(\mathcal{A})$
- ▶ Ένα σύνολο καταστάσεων $states(\mathcal{A})$
 - ▶ Ορισμένες τις ονομάζουμε **αρχικές καταστάσεις** $start(\mathcal{A})$
 - ▶ Ορισμένες τις ονομάζουμε **καταστάσεις τερματισμού** $halt(\mathcal{A})$
- ▶ Μια συνάρτηση αλλαγής κατάστασης $trans(\mathcal{A}) \subseteq states(\mathcal{A}) \times (in(\mathcal{A}) \cup out(\mathcal{A}) \cup int(\mathcal{A})) \times states(\mathcal{A})$
 - ▶ Για κάθε κατάσταση κ και κάθε ενέργεια ϵ
 - ▶ Υπάρχει μια μετάβαση $(\kappa, \epsilon, \kappa') \in trans(\mathcal{A})$



Ορισμός Αυτόματου Εισόδου/Εξόδου (2)

Μια εκτέλεση του αυτόματου εισόδου/εξόδου \mathcal{A} περιγράφεται:

$$k_0, \epsilon_1, k_1, \epsilon_2, \dots, \epsilon_r, k_r, \dots$$

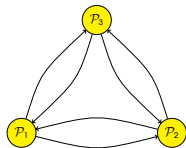
όπου για κάθε $r \geq 0$ ισχύει ότι $(k_r, \epsilon_{r+1}, k_{r+1}) \in \text{trans}(\mathcal{A})$
Σύμφωνα με τα τρία σύνολα από ενέργειες εισόδου $\text{in}(\mathcal{A})$ --
ενέργειες εξόδου $\text{out}(\mathcal{A})$, και εσωτερικές ενέργειες $\text{int}(\mathcal{A})$

- ▶ Ορίζουμε τις **εξωτερικές ενέργειες** $\text{ext}(\mathcal{A}) = \text{in}(\mathcal{A}) \cup \text{out}(\mathcal{A})$
- ▶ Ορίζουμε τις **εσωτερικά ελεγχόμενες ενέργειες** $\text{local}(\mathcal{A}) = \text{out}(\mathcal{A}) \cup \text{int}(\mathcal{A})$
- ▶ Ορίζουμε **όλες τις ενέργειες** $\text{actions}(\mathcal{A}) = \text{in}(\mathcal{A}) \cup \text{out}(\mathcal{A}) \cup \text{int}(\mathcal{A})$



Σύνθεση Αυτόματων Εισόδου/Εξόδου

Παράδειγμα Συστήματος



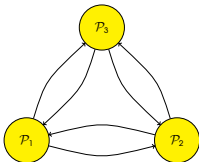
- ▶ Η μοντελοποίηση ενός ασύγχρονου καταναμημένου συστήματος προκύπτει από την **σύνθεση** ενός πλήθους αυτόματων εισόδου/εξόδου
- ▶ Αντιστοιχούμε ένα αυτόματο



Σύνθεση Αυτόματων Εισόδου/Εξόδου

Παράδειγμα Συστήματος

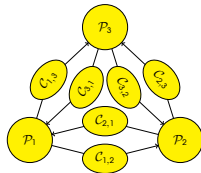
- ▶ Η μοντελοποίηση ενός ασύγχρονου καταναμημένου συστήματος προκύπτει από την **σύνθεση** ενός πλήθους αυτόματων εισόδου/εξόδου
- ▶ Αντιστοιχούμε ένα αυτόματο
 1. σε κάθε διεργασία



Σύνθεση Αυτόματων Εισόδου/Εξόδου

Παράδειγμα Συστήματος

- ▶ Η μοντελοποίηση ενός ασύγχρονου καταναμημένου συστήματος προκύπτει από την **σύνθεση** ενός πλήθους αυτόματων εισόδου/εξόδου
- ▶ Αντιστοιχούμε ένα αυτόματο
 1. σε κάθε διεργασία
 2. σε κάθε κανάλι επικοινωνίας



Μέτρηση πολυπλοκότητας

- ▶ Πολυπλοκότητα Επικοινωνίας
 - ▶ Συνολικός αριθμός μηνυμάτων που στάλθηκαν ή παραλήφθηκαν
- ▶ Χρονική πολυπλοκότητα
 - ▶ Η χρονική απροσδιοριστία δεν μας επιτρέπει να μετρήσουμε με ευκολία
 - ▶ Κάνουμε τις εξής παραδοχές
 1. Θέτουμε ένα άνω φράγμα l στον χρόνο εκτέλεσης κάθε ενέργειας ϵ σε κάποια κατάσταση κ
 2. Θέτουμε ένα άνω φράγμα d στον χρόνο παράδοσης του παλαιότερου μηνύματος που βρίσκεται σε κάποιο κανάλι επικοινωνίας

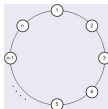


Πρόβλημα Εκλογής Αρχηγού

Πρόβλημα Εκλογής Αρχηγού

Η εκλογή αρχηγού σε ένα δίκτυο απαιτεί την επιλογή μιας μοναδικής διεργασίας που θα βρεθεί στην κατάσταση 'αρχηγός' (ή 'εκλεγμένη') ενώ όλες οι άλλες διεργασίες βρίσκονται στην κατάσταση 'μη-αρχηγός' (ή 'μη εκλεγμένη').

Δίκτυο δακτυλίου



- ▶ Θεωρούμε ότι οι διεργασίες είναι αριθμημένες από το 1 έως το n με δεξιάστροφη κατεύθυνση – οι διεργασίες δεν έχουν γνώση αυτού του τρόπου αρίθμησης
 - ▶ Η διεργασία με τη μεγαλύτερη ταυτότητα είναι η u_{max}
- ▶ Θεωρούμε ότι τα κανάλια επικοινωνίας είναι **αξιόπιστα** και **FIFO**



Ο Αλγόριθμος των LeLann, Chang και Roberts

Αλγόριθμος AsynchLCR

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

- ▶ Ο αλγόριθμος σχεδιάστηκε αρχικά για ασύγχρονα κατανομημένα συστήματα
- ▶ Η ίδια ιδέα λειτουργεί και στα ασύγχρονα συστήματα
- ▶ Προσαρμόζουμε τον αλγόριθμο ως εξής
 - ▶ Τα εξερχόμενα μηνύματα τοποθετούνται σε μια ουρά



Αυτόματο AsynchLCR_U (1)

Ενέργειες:

- ▶ Ενέργειες Εισόδου $in(AsynchLCR_U)$
 1. $receive(\tau)_{u-1,u}$, όπου τ μια ταυτότητα
- ▶ Ενέργειες Εξόδου $out(AsynchLCR_U)$
 1. $send(\tau)_{u,u+1}$, όπου τ μια ταυτότητα
 2. $leader_u$

Καταστάσεις:

- ▶ τ – μια ταυτότητα, αρχικά η ταυτότητα της u
- ▶ $send$ – μια ουρά (FIFO) με ταυτότητες, αρχικά περιέχει μόνο την ταυτότητα της u
- ▶ $status$ – μπορεί να πάρει τιμές {unknown, chosen, reported}, αρχικά unknown.



Μεταβάσεις:

- ▶ $send(\tau)_{u,u+1}$
 - ▶ προϋπόθεση – τ πρώτη στην $send$
 - ▶ αποτέλεσμα – αφαίρεση του πρώτου στοιχείου της $send$
- ▶ $receive(\tau)_{u-1,u}$
 - ▶ αποτέλεσμα
 - $\tau > u$ – τοποθέτησε την τ στην ουρά $send$
 - $\tau = u$ – $status = chosen$
 - $\tau < u$ – τίποτα
- ▶ $leader_u$
 - ▶ προϋπόθεση – $status == chosen$
 - ▶ αποτέλεσμα – $status = reported$



- ▶ Η πολυπλοκότητα επικοινωνίας είναι $O(n^2)$
- ▶ Χρονική πολυπλοκότητα
 - ▶ Ένα μήνυμα μπορεί να καθυστερήσει σε κάποια διεργασία όπου υπάρχουν (το πολύ) n μηνύματα σε αναμονή – εφόσον (το πολύ) l καθυστερήσει ανά μήνυμα, η καθυστέρηση είναι $O(nl)$ ή $O(nd)$ για τα κανάλια επικοινωνίας αντίστοιχα
 - ▶ Εφόσον θα περάσει από όλες τις άλλες διεργασίες και όλα τα κανάλια επικοινωνίας, η χρονική πολυπλοκότητα είναι $O(n^2(l+d))$
 - ▶ Στην πραγματικότητα ο AsynchLCR είναι πιο γρήγορος – εξετάζοντας πιο προσεκτικά μπορούμε να δείξουμε $O(n(l+d))$

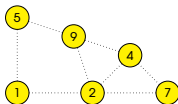


Κατασκευή Επικαλυπτικού Δέντρου (1)

Κατασκευή Επικαλυπτικού Δέντρου

Σε ένα δίκτυο G , οι διεργασίες πρέπει να κατασκευάσουν ένα επικαλυπτικό δέντρο $T(G)$, με ρίζα την διεργασία u_0 .

Γενικό Δίκτυο



- ▶ Μπορούμε να προσαρμόσουμε τον αλγόριθμο SynchBFS στο μοντέλο των ασύγχρονων δικτύων
 - ▶ Ο αλγόριθμος εξακολουθεί να διασφαλίζει την κατασκευή ενός επικαλυπτικού δέντρου
 - ▶ Μπορεί να μην πληροί τις ιδιότητες της αναζήτησης κατά εύρος



Κατασκευή Επικαλυπτικού Δέντρου (2)

Αλγόριθμος AsynchSpanningTree

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.



Αυτόματο AsyncSpanningTree_u (1)

Ενέργειες:

- ▶ Ενέργειες Εισόδου $in(AsyncSpanningTree_u)$
 1. $receive("search")_{u,v}$, όπου $v \in nbrs$
- ▶ Ενέργειες Εξόδου $out(AsyncSpanningTree_u)$
 1. $send("search")_{u,v}$, όπου $v \in nbrs$
 2. $parent(v)_u$, όπου $v \in nbrs$

Καταστάσεις:

- ▶ $parent \in nbrs \cup \{null\}$ -- αρχικά $null$
- ▶ $reported$ -- τύπου $boolean$, αρχικά $false$.
- ▶ για κάθε $v \in nbrs$ -- $send(v) \in \{search, null\}$ -- αρχικά $search$ αν $u = u_0$, αλλιώς $null$



Αυτόματο AsyncSpanningTree_u (2)

Μεταβάσεις:

- ▶ $send("search")_{u,v}$
 - ▶ προϋπόθεση -- $send(v) == search$
 - ▶ αποτέλεσμα -- $send(v) = null$
- ▶ $receive("search")_{u,v}$
 - ▶ αποτέλεσμα -- αν $u \neq u_0$ και $parent == null$ τότε
 $parent = v$
για κάθε $k \in nbrs - v - send(k) = search$
- ▶ $parent(v)_u$
 - ▶ προϋπόθεση -- $parent == v$, $reported == false$
 - ▶ αποτέλεσμα -- $reported = true$



Χαρακτηριστικά του Αλγόριθμου AsyncSpanningTree

- ▶ Ο αλγόριθμος AsyncSpanningTree κατασκευάζει ένα επικαλυπτικό δέντρο
 - ▶ Η απόσταση μιας διεργασίας από την u_0 δεν είναι η ίδια στο G και στο $T(G)$
- ▶ Η πολυπλοκότητα επικοινωνίας είναι $\mathcal{O}(m)$
- ▶ Χρονική πολυπλοκότητα:
 - ▶ Δεν παρατηρούνται συνωσιμοί μηνυμάτων
 - ▶ Όλες οι διεργασίες θα έχουν επιλέξει γονέα εντός χρόνου $\delta(l + d) + l$

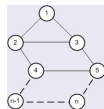


Αναζήτηση κατά Εύρος

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 $T(G)$.

Γενικό Δίκτυο



- ▶ Στα σύγχρονα συστήματα χρησιμοποιήσαμε τον αλγόριθμο SychBFS
- ▶ Μπορούμε να μεταβάλουμε τον AsyncSpanningTree έτσι ώστε οι διεργασίες να διορθώνουν 'λανθασμένους' γονείς
- ▶ Εάν μια διεργασία λάβει πληροφορία από ένα γείτονα που είναι πιο κοντά από τον γονέα της, τότε αλλάζει γονέα



Αλγόριθμος AsynchBFS

Κάθε διεργασία u διατηρεί μια μεταβλητή d_u όπου αποθηκεύει την απόσταση της από την u_0 (σύμφωνα με τις τρέχουσες συνθήκες), αρχικά αν $u \neq u_0$, $d_u = \infty$ αλλιώς αν $u = u_0$, $d_u = 0$. Αρχικά, η διεργασία u_0 στέλνει την τιμή της d_{u_0} σε όλους τους γείτονες της. Σε κάθε γύρο, εάν μια διεργασία λάβει ένα μήνυμα m από την v όπου $m + 1 < d_u$, θέτει $d_u = m + 1$, και την μεταβλητή **γονέας** με την ταυτότητα της v από όπου έλαβε το μήνυμα.

- ▶ Έστω $d(u)$ η απόσταση της u_0 από την u στο G
- ▶ Κατά την εκτέλεση του αλγορίθμου, για οποιαδήποτε γειτονικές u, v είτε θα ισχύει ότι $d_v < d_u + 1$ είτε το d_u στέλνεται από την u στη v



Ενέργειες:

- ▶ Ενέργειες Εισόδου $in(AsynchSpanningTree_u)$
 1. $receive(m)_{u,v}$, όπου $m \in \mathcal{N}, v \in nbrs$
- ▶ Ενέργειες Εξόδου $out(AsynchSpanningTree_u)$
 1. $send(m)_{u,v}$, όπου $m \in \mathcal{N}, v \in nbrs$

Καταστάσεις:

- ▶ $d_u \in \mathcal{N} \cup \{\infty\}$ – αρχικά 0 αν $u = u_0$ αλλιώς ∞
- ▶ $parent \in nbrs \cup \{\text{null}\}$ – αρχικά null
- ▶ για κάθε $v \in nbrs - send(v)$ – μια ουρά (FIFO) στοιχείων του \mathcal{N} , αρχικά περιέχει το στοιχείο 0, αν $u = u_0$, αλλιώς είναι άδεια



Μεταβάσεις:

- ▶ $send(m)_{u,v}$
 - ▶ προϋπόθεση – m πρώτο στην ουρά $send(v)$
 - ▶ αποτέλεσμα – αφαίρεση του πρώτου στοιχείου της $send(v)$
- ▶ $receive(m)_{u,v}$
 - ▶ αποτέλεσμα – αν $m + 1 < d_u$ τότε
 - $parent = v$
 - για κάθε $k \in nbrs - v$ – τοποθέτησε το d_k στην ουρά $send(k)$



- ▶ Σε κάθε χρονική στιγμή της εκτέλεσης όπου κάποια d_u δεν είναι άπειρη, η τιμή της d_u θα είναι το μήκος κάποιου μονοπατιού από την u_0 στην u
 - ▶ $d(u) \leq d_u < n$
 - ▶ η μεταβλητή d_u αλλάζει τιμή το πολύ n φορές
- ▶ Η πολυπλοκότητα επικοινωνίας είναι $\mathcal{O}(nm)$



Λήμμα

Για κάθε u εντός χρόνου $d(u)n(l) + (d)$ θα ισχύει $d_u = d(u)$

- ▶ Για $d(u) = 0$ είναι προφανές
- ▶ Έστω οπ ισχύει για κάθε v όπου $d(v) \leq k$
- ▶ Έστω η διεργασία u με $d(u) = k + 1$ και διεργασία v (γειτονική της u) με $d(v) = k$
- ▶ Εντός χρόνου $kn(l) + (d)$ η v έθεσε $d(v) = k$ και αποφάσισε να στείλει k στην u
- ▶ Εντός επιπλέον χρόνου $n(l)$ η v θα στείλει το k στο C_{vu}
- ▶ Εντός επιπλέον χρόνου $n(d)$ η u θα το λάβει, θα θέσει $d_u = k + 1$ και θα επιλέξει ως γονέα της την v



Θεώρημα

Με την εκτέλεση του AsynchBFS το σύστημα συγκλίνει σε μια κατάσταση όπου έχει κατασκευαστεί επικαλυπτικό δένδρο $T(G)$ τέτοιο ώστε η απόσταση μιας κορυφής από την u_0 είναι η ίδια και στο G και στο $T(G)$ και αυτό ολοκληρώνεται σε χρόνο $\mathcal{O}(\delta n(l) + (d))$

- ▶ Η τεχνική της 'σύγκλισης' είναι χαρακτηριστική στα ασύγχρονα κατανεμημένα συστήματα
- ▶ Ίδια τεχνική χρησιμοποιείται και στον αλγόριθμο BellmanFord για εύρεση συντομότερων μονοπατιών



Σύνοψη 5^{ης} Διάλεξης

Προηγούμενο Μάθημα

Προηγούμενο Μάθημα
Βυζαντινά Σφάλματα

Ασύγχρονα Κατανεμημένα Συστήματα

Συζήτηση
Μοντελοποίηση Συστήματος
Βασικοί Κατανεμημένοι Αλγόριθμοι

Σύνοψη Μαθήματος

Σύνοψη Μαθήματος
Βιβλιογραφία
Επόμενο Μάθημα



Σύνοψη Μαθήματος

- ▶ Ασύγχρονα Κατανεμημένα Συστήματα
 - ▶ Μοντελοποίηση
 - ▶ Αυτόματα Εισόδου / Εξόδου
- ▶ Βασικοί Κατανεμημένοι Αλγόριθμοι
 - ▶ Εκλογή Αρχηγού
 - ▶ Κατασκευή Επικαλυπτικών Δέντρων
 - ▶ Αναζήτηση κατά Εύρος



Βιβλιογραφία (1)

- ▶ Τόμος I από τις Πανεπιστημιακές Σημειώσεις "Θεμελιώδη Ζητήματα Κατανεμημένων Συστημάτων" (Π.Σπυράκης, Β.Ταμπακάς):
 1. Κεφάλαιο 3: Βασικά πρωτόκολλα Εκλογής Αρχηγού
- ▶ Βιβλίο "Distributed Algorithms" (N.Lynch)
 1. Κεφάλαιο 8: Modelling II: Asynchronous System Model – Μόνο 8.1, 8.2
 2. Κεφάλαιο 14: Modelling IV: Asynchronous Network Model
 3. Κεφάλαιο 15: Basic Asynchronous Network Algorithms
- ▶ Βιβλίο "Distributed Computing Fundamentals, Simulations, and Advanced Topics" (H.Atliya, J.Welch)
 1. Κεφάλαιο 2: Basic Algorithms in Message Passing Systems
 2. Κεφάλαιο 3: Leader Election in Rings



Βιβλιογραφία (2)

- ▶ Βιβλίο "Introduction to Distributed Algorithms" (G.Tel)
 1. Κεφάλαιο 2: The Model
 2. Κεφάλαιο 6: Wave and Traversal Algorithms
 3. Κεφάλαιο 7: Election Algorithms
- ▶ Βιβλίο "Distributed Systems, Concepts and Design" (G.Coulouris, J.Dollimore, T.Kindberg)
 1. Κεφάλαιο 2: System Models
 2. Κεφάλαιο 11: Coordination and Agreement
- ▶ Βιβλίο "Distributed Systems: Principles and Paradigms" (A.Tanenbaum, M.Steen)
 1. Κεφάλαιο 5: Synchronization – Μόνο 5.4



Επόμενο Μάθημα

- ▶ Ασύγχρονα Κατανεμημένα Συστήματα
- ▶ Λογικός Χρόνος
- ▶ Αμοιβαίος Αποκλεισμός

