Κεφάλαιο 7

Αμοιβαίος Αποκλεισμός και Εκλογή Αρχηγού

Σύνοψη
Ο αμοιβαίος αποκλεισμός εμφανίζεται στα κατανεμημένα συστήματα επειδή εν υπάρχει αξιόπιστη πληροφόρηση για την καθολική κατάσταση του συστήματος, ούτε κοινό ρολόι μεταξύ των επικοινωνούντων συστημάτων. Στο κεφάλαιο εξετάζεται ο αμοιβαίος αποκλεισμός από την πρόσβαση σε ένα απομακρυσμένο πόρο, όπου κανένα σύστημα πελάτη δεν έχει πρόσβαση και το σύστημα μένει σε μια μπλοκαρισμένη και αδρανή κατάσταση. Παρουσιάζονται τρεις αλγόριθμοι για την αποφυγή αδιεξόδων, ο αλγόριθμος του Raymond ο αλγόριθμος των Ricart και Agrawala και ο ο αλγόριθμος του Maekawa. Επίσης, σε ένα κατανεμημένο σύστημα πολλές φορές απαιτείται η εκλογή κάποιου κόμβου ως αρχηγού. Παρουσιάζονται διάφοροι αλγόριθμοι εκλογής αρχηγού, όπως π.χ. του LeLann, των Chang και Roberts, με αντίστοιχα παραδείγματα.

Προαπαιτούμενη Γνώση

1) Andrew S. Tanenbaum, Herbert Bos (2018), Σύγχρονα Λειτουργικά Συστήματα, Έκδοση: 4η Αμερικανική, Εκδόσεις Κλειδάριθμος ΕΠΕ

2) ANDREW S. TANENBAUM, DAVID J. WETHERALL, ΔΙΚΤΥΑ ΥΠΟΛΟΓΙΣΤΩΝ, Έκδοση: 5η Αμερικανική/2011, Εκδόσεις Κλειδάριθμος ΕΠΕ

3) Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, Εισαγωγή στους Αλγορίθμους, Έκδοση: 1η/2016, ΙΔΡΥΜΑ ΤΕΧΝΟΛΟΓΙΑΣ & ΕΡΕΥΝΑΣ-ΠΑΝΕΠΙΣΤΗΜΙΑΚΕΣ ΕΚΔΟΣΕΙΣ ΚΡΗΤΗΣ

7.1 Αμοιβαίος αποκλεισμός

Στον τομέα της επιστήμης των υπολογιστών, είναι μείζονος σημασίας η ύπαρξη ενός μηχανισμού διαχείρισης συγκρούσεων στην περίπτωση, όπου ταυτόχρονες διεργασίες φτάνουν σε μία βασική (κοινή) περιοχή πρόσβασης. Για τον λόγο αυτό υπάρχει η έννοια του αμοιβαίου αποκλεισμού, που εξυπηρετεί την αποκλειστική πρόσβαση στη συγκεκριμένη βασική περιοχή. Ο αμοιβαίος αποκλεισμός είναι μία έννοια που μπορεί να σταθεί όπου υπάρχει κοινή χρήση πόρων, όπως π.χ ως τοποθεσία κοινής μνήμης. Έτσι, όταν δύο παράλληλες διεργασίες επιθυμούν να αποκτήσουν πρόσβαση σε έναν κοινόχρηστο πόρο, δεν θα πρέπει να υπάρχει διαφωνία επεξεργασίας, καθώς θα πρέπει να επιβάλλεται ο αμοιβαίος αποκλεισμός, δηλαδή ο αποκλεισμός μιας διεργασίας από μία ενέργεια που εκτελεί ταυτόχρονα κάποια άλλη διεργασία.

Σε γενικές γραμμές ο αμοιβαίος αποκλεισμός απαιτεί μία μόνο διεργασία να εκτελεί λειτουργίες σε κοινούς πόρους. Σε επόμενο στάδιο εάν υπάρχουν δύο διεργασίες, όπου και οι δύο θα θελήσουν να αποκτήσουν πρόσβαση σε κοινό πόρο, τότε η μία διεργασία θα πρέπει να περιμένει τη δεύτερη. Αυτή η λειτουργία όμως δημιουργεί αρκετά προβλήματα κυρίως στο ζήτημα του χρόνου εκτέλεσης, για αυτό τον λόγο προτείνεται η αποτροπή πρόσβασης σε διαμοιραζόμενους πόρους από πολλές διεργασίες την ίδια χρονική στιγμή. Ο στόχος είναι να παρέχονται μέθοδοι ή αλγόριθμοι, οι οποίοι να μπορούν να ικανοποιήσουν αυτή την ανάγκη.

Ο αμοιβαίος αποκλεισμός συναντάται σαν έννοια και στα συστήματα κεντροποιημένων υπολογιστών, αλλά και στα κατανεμημένα συστήματα.

Σε ένα σύστημα υπολογιστή, η μνήμη και άλλοι πόροι διαμοιράζονται μεταξύ διαφορετικών διεργασιών. Η κατάσταση που βρίσκονται οι πόροι που χρησιμοποιούνται από την ίδια ομάδα διεργασιών, όπως και η κατάσταση των χρηστών είναι εύκολα διαθέσιμα στην κοινόχρηστη μνήμη, έτσι με τη βοήθεια μιας κοινής μεταβλητής (για παράδειγμα: Semaphores) το πρόβλημα αμοιβαίου αποκλεισμού μπορεί εύκολα να λυθεί.

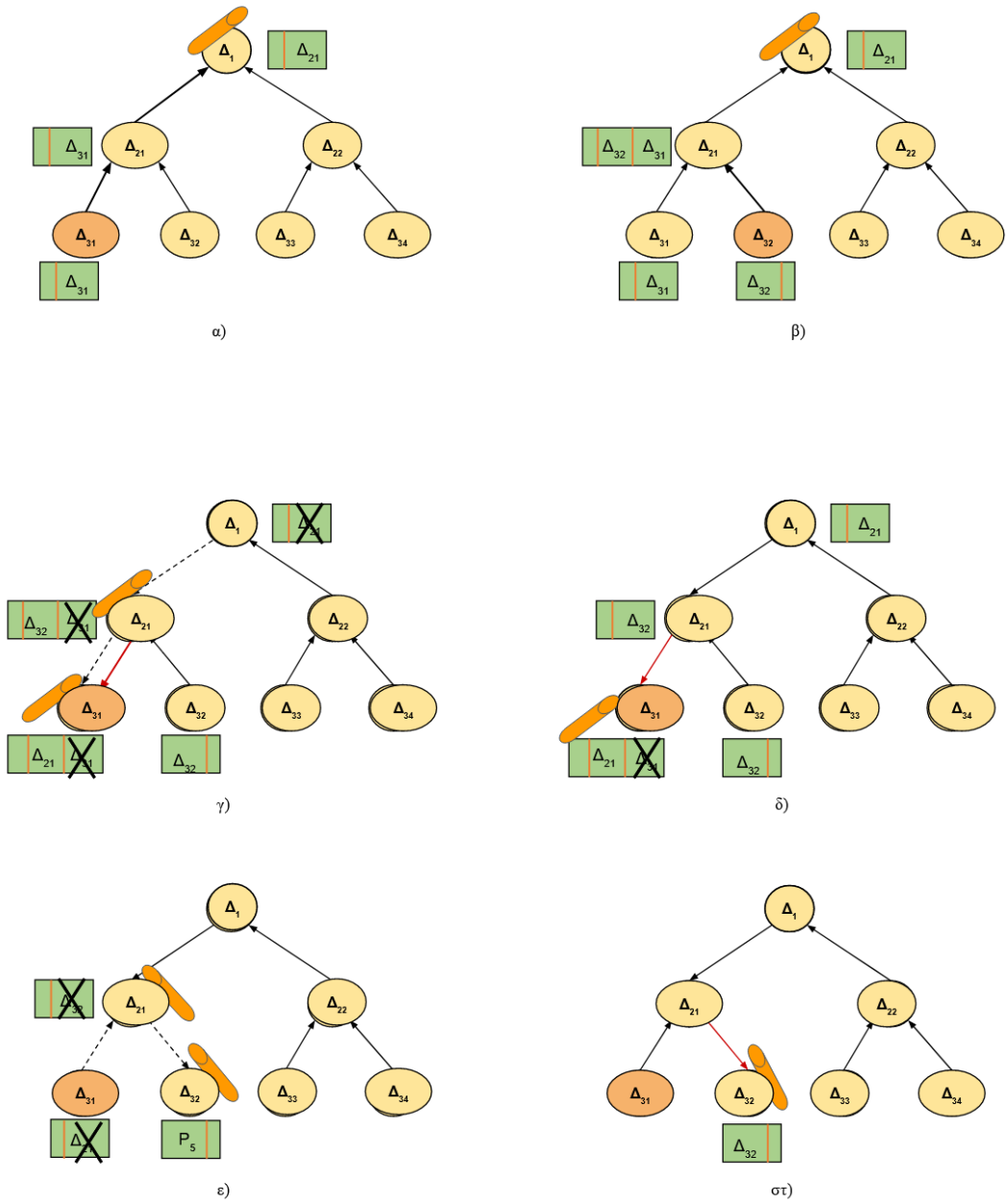
Οι σημαφόροι είναι ακέραιες μεταβλητές, που μοιράζονται μεταξύ πολλών διεργασιών. Ο κύριος στόχος της χρήσης ενός σημαφόρου είναι ο συγχρονισμός διεργασιών και ο έλεγχος πρόσβασης για έναν κοινό πόρο σε ένα ταυτόχρονο περιβάλλον. Η αρχική τιμή ενός σημαφόρου εξαρτάται από το πρόβλημα. Συνήθως χρησιμοποιείται σαν αρχική τιμή ο αριθμός των διαθέσιμων πόρων. Σε γενικές γραμμές είναι ένα εργαλείο συγχρονισμού που ΓΟΝΕΑ του κόμβου ξεκινάει και η όποια ενέργεια. Το δικαίωμα εισόδου το έχει σε αυτή την περίπτωση μόνο μία διεργασία, που ονομάζεται κάτοχος (holder) της σκυτάλης. Η συγκεκριμένη διεργασία έχει προκαθορισμένη θέση και βρίσκεται στη ρίζα του δέντρου.

Ο ρόλος της σκυτάλης είναι να ξεκινάει από τη ρίζα του δέντρου, έπειτα να περνάει από τις ακμές του δέντρου στο μονοπάτι με σκοπό να βρει τη διεργασία που έκανε το αίτημα. Κατά τη διάρκεια αυτής της διαδρομής της σκυτάλης και κατά μήκος του μονοπατιού, η κατεύθυνση των ακμών αλλάζει, έτσι ώστε η διεργασία που ζήτησε τη σκυτάλη να βρίσκεται στη νέα ρίζα. Είναι απαραίτητο κάθε διεργασία να διατηρεί μια ουρά, όπου βρίσκονται οι αιτήσεις των διεργασιών, οι οποίες δεν έχουν λάβει ακόμα τη σκυτάλη, καθώς και ο δείκτης μονοπατιού, ο οποίος δείχνει στη διεργασία προς τα πού είναι η ρίζα.

Όπως παρουσιάζεται στην Εικόνα 7.2, η σκυτάλη βρίσκεται στην διεργασία p1, και η διεργασία p4 επιθυμεί να εισέλθει στην κρίσιμη περιοχή της p1. Γι αυτό τον λόγο γίνεται αρχικά η αίτηση στην ουρά της. Επειδή όμως η ουρά της είναι άδεια γιατί είμαστε στην αρχή της διαδικασίας, έχουμε την αποστολή ενός μηνύματος στην p2. Η διεργασία p2 με τη σειρά της λαμβάνει, αποθηκεύει στην ουρά της και προωθεί το μήνυμα αυτό στην p1. Η p1 εισάγει το μήνυμα αυτό στην ουρά της.

Στο επόμενο βήμα φαίνεται η p5 να επιθυμεί να εισέλθει και αυτή στην κρίσιμη περιοχή της, οπότε εισάγει την αίτησή της στην ουρά της. Έπειτα πρέπει να γίνει αποστολή μηνύματος προς τη διεργασία που δείχνει ο δείκτης, δηλαδή προς την p2. Η p2 όπως και στη προηγούμενη περίπτωση αποθηκεύει την αίτηση της p5 στην ουρά της. Εδώ τώρα αντιμετωπίζουμε την περίπτωση όπου η ουρά της p2 είναι ήδη γεμάτη, όταν φτάνει το μήνυμα από την p5. Για αυτό τον λόγο η p2 δεν προωθεί το μήνυμα αυτό προς τη ρίζα.

Όταν η p1 βγει από την κρίσιμη περιοχή της, διαγράφει την αίτηση της p2 από την ουρά της, και της στέλνει τη σκυτάλη (ενημερώνοντας κατάλληλα τους δείκτες του δέντρου). Η σκυτάλη φτάνει στην p2 , η οποία είχε στην ουρά της το της p5 (μήνυμα που δεν έχει ακόμα μεταβιβαστεί), και στέλνει τη σκυτάλη και το μήνυμα της p5 στην p4. Η p4 διαγράφει την αίτησή της από την ουρά της, που σημαίνει ότι ουσιαστικά έχει μόνο την αίτηση που έλαβε από τη διεργασία p2 στο προηγούμενο βήμα. Όταν η p4 ολοκληρώσει τις ενέργειές της στην κρίσιμη περιοχή της, ελέγχει και εντοπίζει το αίτημα που προέρχεται από την p2. Έτσι η p2 παίρνει τη σκυτάλη από την p4, που σημαίνει ότι η p4 αλλάζει κατεύθυνση στον δείκτη της. Τέλος η διεργασία p5 αποκτά τη σκυτάλη, και μετά την έξοδο από την κρίσιμη περιοχή της δεν την προωθεί, για τον λόγο ότι η ουρά της είναι άδεια μετά τη διαγραφή της δικής της αίτησης.



Εικόνα 7.2 Παράδειγμα του Αλγόριθμου Raymond

7.2.2 Ο αλγόριθμος των Ricart & Agrawala

Ο αλγόριθμος των Ricart & Agrawala είναι κατανεμημένος και σαν γενικό χαρακτηριστικό του, απαιτεί την ολική και όχι την μερική διάταξη των γεγονότων που αφορούν σε ένα κατανεμημένο σύστημα. Σύμφωνα με τον αλγόριθμο αυτό, στην περίπτωση που κάποια διεργασία εισέρχεται στην κρίσιμη περιοχή στέλνει μήνυμα στις υπόλοιπες διεργασίες. Κατόπιν, όταν κάποια διεργασία παραλήπτης παραλαμβάνει τέτοιο μήνυμα κάνει μία σειρά ενεργειών ανάλογα με την κατάστασή της. Πιο συγκεκριμένα, η διεργασία παραλήπτης στην περίπτωση που δεν θέλει να εισέλθει στην κρίσιμη περιοχή και, φυσικά, δεν βρίσκεται σε αυτή απαντά θετικά. Βέβαια στην περίπτωση που η διεργασία παραλήπτης βρίσκεται στην

κρίσιμη περιοχή τότε ή δεν στέλνει απάντηση (εφόσον ξεπεραστεί το όριο χρόνου αναμονής που καθορίζεται από τον αποστολέα) ή απαντά αρνητικά. Στην περίπτωση κατά την οποία η διεργασία παραλήπτης δεν βρίσκεται στην κρίσιμη περιοχή, αλλά επιθυμεί να μπει στην κρίσιμη περιοχή, τότε πρέπει να συγκρίνει τις χρονοσφραγίδες και να απαντήσει αναλόγως. Έτσι, στην περίπτωση που κάποια διεργασία αποστολέας καταφέρει και μαζέψει όλες τις θετικές απαντήσεις, τότε μπορεί να εισέλθει στην κρίσιμη περιοχή.

7.2.3 Ο αλγόριθμος του Maekawa

Ο αλγόριθμος του Maekawa είναι μια προσέγγιση βασισμένη στην απαρτία (Quorum based) για τη διασφάλιση από το πρόβλημα του αμοιβαίου αποκλεισμού στα κατανεμημένα συστήματα. Στους αλγορίθμους που βασίζονται στην άδεια (permission-based algorithms), όπως είναι ο αλγόριθμος του Lamport, ο αλγόριθμος των Ricart-Agrawala κ.ά., μια τοποθεσία ζητά άδεια από κάθε άλλη τοποθεσία προκειμένου να μπορεί να εκτελέσει στην κρίσιμη περιοχή. Στην προσέγγιση που βασίζεται στην απαρτία, μια τοποθεσία δεν ζητά άδεια από κάθε άλλη τοποθεσία, αλλά από ένα υποσύνολο τοποθεσιών που ονομάζεται απαρτία. Η απαρτία ουσιαστικά αποτελεί ένα αντιπροσωπευτικό υποσύνολο όλων των κόμβων.

Σε αυτόν τον αλγόριθμο:

- Χρησιμοποιούνται τρεις τύποι μηνυμάτων (REQUEST, REPLY και RELEASE).
- Μια τοποθεσία στέλνει ένα μήνυμα REQUEST σε όλες τις άλλες τοποθεσίες του συνόλου της απαρτίας, για να λάβει την άδειά τους να εισέλθει στην κρίσιμη περιοχή.
- Μια τοποθεσία στέλνει ένα μήνυμα ΑΠΑΝΤΗΣΗ στην τοποθεσία που υπέβαλε το αίτημα για να δώσει την άδειά της να εισέλθει στο κρίσιμο τμήμα.
- Μια τοποθεσία στέλνει μήνυμα RELEASE σε όλες τις άλλες τοποθεσίες του συνόλου της απαρτίας κατά την έξοδο από το κρίσιμο τμήμα.

7.3 Αλγόριθμοι εκλογής αρχηγού

Οι κόμβοι, όπως έχει αναφερθεί, επικοινωνούν μεταξύ τους χρησιμοποιώντας κοινόχρηστες μνήμες ή μέσω διαβίβασης μηνυμάτων. Για την αποτελεσματική εκτέλεση οποιασδήποτε κατανεμημένης εργασίας, η βασική απαίτηση από την πλευρά των κόμβων είναι ο συντονισμός. Σε ένα καθαρά κατανεμημένο σύστημα, δεν υπάρχει κανένας κεντρικός κόμβος ελέγχου που να συντονίζει τις αποφάσεις. Αυτό σημαίνει ότι κάθε κόμβος πρέπει να επικοινωνεί με τους υπόλοιπους κόμβους του δικτύου για να λάβει τη σωστή απόφαση που αφορά στην εκτέλεση των διεργασιών. Ο προσδιορισμός ενός μεμονωμένου κόμβου ως οργανωτή στα κατανεμημένα συστήματα είναι ένα δύσκολο ζήτημα που απαιτεί κατάλληλα σχεδιασμένους εκλογικούς αλγόριθμους εκλογής αρχηγού/οργανωτή.

Συχνά κατά τη διαδικασία λήψης αποφάσεων, δεν παίρνουν όλοι οι κόμβοι την ίδια απόφαση, επομένως η επικοινωνία μεταξύ των κόμβων άρα και η διαδικασία λήψης αποφάσεων γίνεται χρονοβόρα. Επομένως, ένα ζητούμενο είναι η συνεννόηση και η συνέπεια στις αποφάσεις από όλους τους κόμβους για την εκτέλεση στην ίδια κρίσιμη περιοχή. Οι κεντρικοί κόμβοι ελέγχου μπορεί να επιλεγούν από την ομάδα των διαθέσιμων κόμβων προς μείωση της πολυπλοκότητας της λήψης των αποφάσεων.

Πολλοί κατανεμημένοι αλγόριθμοι απαιτούν έναν κόμβο για να λειτουργεί ως συντονιστής, εκκινητής, ή με κάποιον άλλον ειδικό ρόλο. Η εκλογή αρχηγού είναι μια τεχνική που μπορεί να βοηθήσει σε αυτό το ζήτημα, προκειμένου να προσδιοριστεί ένας κεντρικός κόμβος ελέγχου σε ένα κατανεμημένο σύστημα. Ο κόμβος αυτός συνήθως εκλέγεται από την ομάδα των κόμβων ως ηγέτης για να χρησιμεύσει ως ο κεντρικός ελεγκτής για αυτό τον κατανεμημένο σύστημα.

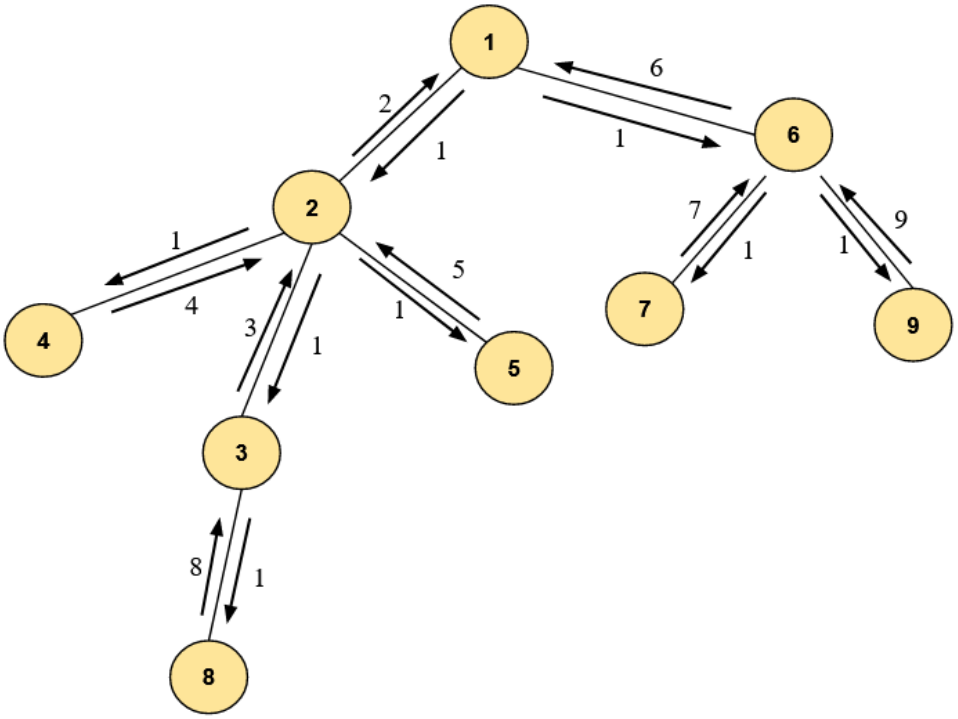
Σε κάθε αλγόριθμο εκλογής αρχηγού, επιλέγεται αρχηγός με βάση κάποιο κριτήριο, όπως η επιλογή για παράδειγμα του κόμβου με το μεγαλύτερο ή μικρότερο αναγνωριστικό [εικόνα 3]. Μόλις εκλεγεί ο αρχηγός, οι κόμβοι φτάνουν σε μια συγκεκριμένη κατάσταση που είναι γνωστή ως τερματική κατάσταση.

Στους αλγόριθμους εκλογής αρχηγού :

- Οι κρίσιμες περιοχές χωρίζονται σε εκλεγμένες και μη εκλεγμένες.
- Όταν ένας κόμβος εισέρχεται σε μία οποιαδήποτε κατάσταση, πάντα παραμένει σε αυτή την κατάσταση.
- Κάθε αλγόριθμος εκλογής αρχηγού πρέπει να ικανοποιεί τις συνθήκες ασφάλειας και λειτουργικότητας για να είναι δυνατή η εκτέλεσή του.
- Η προϋπόθεση ασφαλείας για την εκλογή αρχηγού απαιτεί ότι μόνο ένας κόμβος μπορεί να εισέλθει στην εκλεγμένη κατάσταση (να εκλεγεί αρχηγός) και τελικά να γίνει ο ηγέτης του κατανεμημένου συστήματος.

Οι πληροφορίες ανταλλάσσονται μεταξύ των κόμβων μεταδίδοντας μηνύματα από τον ένα στον άλλο μέχρι να επιτευχθεί συμφωνία. Μόλις ληφθεί μια απόφαση, ένας κόμβος εκλέγεται ως αρχηγός και όλοι οι άλλοι κόμβοι θα αναγνωρίσουν τον ρόλο αυτού του κόμβου ως αρχηγού.

Στην εικόνα 3 παρουσιάζεται ένα απλό παράδειγμα της βασικής λειτουργίας των αλγορίθμων εκλογής αρχηγού. Το τελικό ζήτημα είναι να ανακηρυχθεί αρχηγός ο κόμβος 1. Επιλέχθηκε ο κόμβος 1 καθαρά λόγω του ονόματός του, επειδή έχει τον μικρότερο αριθμό. Πιο αναλυτικά, οι κόμβοι “φύλλα” του τελευταίου επιπέδου του δέντρου στέλνουν τον αριθμό τους στους από πάνω κόμβους. Οι από πάνω κόμβοι παίρνουν τον αριθμό των από κάτω και τον συγκρίνουν με τον δικό τους. Εάν είναι μεγαλύτερος τον αγνοούν και στέλνουν το δικό τους νούμερο, αν είναι μικρότερος τότε προωθούν τον αριθμό του από κάτω επιπέδου. Όταν φτάσουμε στον κόμβο 1 που πληροί τα κριτήρια (μικρότερος αριθμός), ο κόμβος 1 αποφασίζει να εκλεχθεί αρχηγός. Έπειτα ο κόμβος 1 στέλνει μήνυμα να ενημερώσει όλους τους υπόλοιπους κόμβους ότι εκλέχθηκε αρχηγός. Έτσι ο κόμβος 1 στέλνει μηνύματα με τον αριθμό του, ο οποίος πηγαίνει από επίπεδο σε επίπεδο και συγκρίνεται με τους αριθμούς που αντιστοιχούν στους άλλους κόμβους. Οι άλλοι κόμβοι τον δέχονται και καταλαβαίνουν ότι είναι ο αρχηγός.



Εικόνα 7.3 Παράδειγμα εκλογής αρχηγού

7.3.1 Αλγόριθμος Δένδρου εκλογής αρχηγού

Ο αλγόριθμος δένδρου είναι ένας από τους αλγορίθμους που μπορούν να λύσουν το ζήτημα της εκλογής ενός κόμβου ως αρχηγό. Στον αλγόριθμο Δένδρου, η βασική διαφοροποίηση σε σχέση με τους υπόλοιπους είναι ότι υποθέτουμε ότι η τοπολογία των κόμβων είναι είτε δέντρο είτε οποιαδήποτε άλλη τοπολογία η οποία έχει διαθέσιμο κάποιο γενετικό δέντρο.

Θα υποθέσουμε ότι ο αλγόριθμος ξεκινά από κάθε φύλλο του δέντρου. Κάθε διεργασία στον αλγόριθμο είναι υπεύθυνη για τη μετάδοση ακριβώς ενός μηνύματος. Σε περίπτωση που μια διεργασία έχει λάβει ένα μήνυμα από όλους τους γείτονές της εκτός από έναν, η διαδικασία θα στείλει ένα μήνυμα στον συγκεκριμένο γείτονα που δεν έχει στείλει ακόμα μήνυμα. Μια απόφαση λαμβάνεται από μια διεργασία, αφού λάβει μηνύματα από όλες τις γειτονικές διεργασίες της.

Η εκλογή αρχηγού υλοποιείται χρησιμοποιώντας την προσέγγιση κυματοδένδρου (wave tree algorithm), εφόσον η τοπολογία του δικτύου είναι δέντρο ή εάν είναι διαθέσιμο (σε οποιαδήποτε αυθαίρετη τοπολογια) ένα γενητικό δέντρο. Είναι σημαντικό για τη λειτουργία του αλγόριθμου δέντρου να χρησιμοποιούνται, τουλάχιστον, όλα τα φύλλα του δένδρου ως αρχικοποιητές. Μια φάση αφύπνισης (wakeup phase) εισάγεται στον αλγόριθμο μόνο σε περίπτωση, που μέρος των διεργασιών είναι αρχικοποιητές. Αυτό επιτρέπει στον αλγόριθμο να προχωρήσει ακόμη και σε αυτό το σενάριο. Η διαδικασία που είναι υπεύθυνη για την έναρξη της εκλογής θα στείλει πρώτα ένα μήνυμα αφύπνισης στο δέντρο πριν επιχειρήσει να επικοινωνήσει με τις άλλες διεργασίες. Η λογική μεταβλητή *ws* μπορεί να χρησιμοποιηθεί για να διασφαλιστεί ότι μια διεργασία θα στείλει ένα μήνυμα αφύπνισης μόνο μία φορά και η μεταβλητή *wr* μπορεί να χρησιμοποιηθεί για την παρακολούθηση του αριθμού των μηνυμάτων αφύπνισης που έχει λάβει μια διεργασία. Όταν μια διεργασία έχει λάβει ένα μήνυμα αφύπνισης από κάθε επικοινωνιακό κανάλι -πράγμα

που σημαίνει ότι έχει λάβει και από όλους τους γείτονές της - τότε, ξεκινάει έναν εσωτερικό αλγόριθμο που καλύπτει το δέντρο για να προσδιορίσει τη μικρότερη ταυτότητα. Αυτό διασφαλίζει ότι κάθε διαδικασία επιλέγει την ταυτότητα με το μικρότερο αναγνωριστικό μέσα στο δέντρο για να αναλάβει το ρόλο του αρχηγού. Εάν αυτή η ταυτότητα είναι ίδια με την δικιά της ταυτότητά, τότε κερδίζει τις εκλογές και γίνεται αρχηγός. Σε διαφορετική περίπτωση δεν θα εκλεγεί αρχηγός.

Στον κώδικα 7.1 παρουσιάζεται ο αλγόριθμος δέντρου σε μορφή ψευδοκώδικα. Το παρακάτω κομμάτι κώδικα περιέχει μεταβλητές που σχετίζονται με μια διεργασία που ονομάζεται d, η οποία είναι υπεύθυνη για την εκτέλεση του αλγορίθμου.

```
var ws: boolean                 init false;
    wr: integer                 init 0;
    rec[p]: Boolean for each p ∈ Γείτονικός Κόμβος init false;
    v: D                        init d;
state: (sleep, leader, lost) init sleep;

begin if d is initiator then
    begin ws:=true;
        forall p ∈ Γείτονικός Κόμβος do send <αφυπνίσου> to p
    end;
    while wr < #ΓείτονικόςΚόμβος do
        begin receive <αφυπνίσου>;
            wr:=wr+1;
            if not ws then
                begin ws:=true;
                    forall p ∈ Γείτονικός Κόμβος do send <αφυπνίσου> to p
                end
        end;
/*ξεκινάει ο αλγόριθμος δένδρου*/
    while #{p: ¬rec[p]} >1 do
        begin receive <tok,r> from p;
            rec[p]:=true;
            v:=min(v,r)
        end;
    send <tok, v> to p0 with ¬rec[p0];
    receive <tok,r> from p0;
```

v:=min(v,r);

if v=d then state:=leader else state:=lost;

forall p ∈ Γ ε ί τ ο ν ι κ ός Κ όμ β ο ς, p≠p0 do send <tok,v> to p

end

Κώδικας 7.1: Αλγόριθμος εκλογής αρχηγού σε μορφή δένδρου (FOOTNOTE)

Ο παραπάνω αλγόριθμος εκλογής αρχηγού (κώδικας 7.1) έχει μορφή δέντρου και είναι σε μορφή ψευδοκώδικα. Με p ορίζεται η διεργασία που τρέχει τον αλγόριθμο, με N ορίζεται ο αριθμός των κόμβων και με Y το ύψος του δέντρου.

Εφόσον το πλήθος των ακμών στο δέντρο είναι N-1, η πολυπλοκότητα των μηνυμάτων είναι 4N-4 δεδομένου ότι από κάθε ακμή του δέντρου στέλνονται δύο και δύο μηνύματα δηλαδή O(N). Από τη στιγμή που η πρώτη διεργασία του αλγορίθμου ξεκινάει, όλες οι διεργασίες έχουν στείλει μηνύματα, έτσι μέσα σε Y+1 χρονικές μονάδες όλες οι διεργασίες έχουν ξεκινήσει να εκτελούν τον αλγόριθμο. Αυτό σημαίνει ότι ο μέγιστος χρόνος για να ληφθεί η πρώτη απόφαση είναι Y χρονικές μονάδες μετά την εκκίνηση του αλγορίθμου και η τελευταία το πολύ Y χρονικές μονάδες μετά την πρώτη απόφαση. Αυτό σημαίνει ότι συνολικά έχουμε 3Y+1 χρονικές μονάδες, δηλαδή O(Y).

7.3.2 Αλγόριθμος του Awerbuch

Στον αλγόριθμο του Awerbuch αναφερόμαστε σε τρεις αλγορίθμους - πρωτόκολλα (συγχρονιστής α, β και γ), τα οποία αναπαριστούν ένα όσο το δυνατόν καλύτερα συγχρονισμένο περιβάλλον πάνω από ένα μη συγχρονισμένο δίκτυο. Ο αλγόριθμος του συγχρονιστή-α είναι αρκετά γρήγορος ως προς την υλοποίησή του, με μειονέκτημα την αποστολή μεγάλου αριθμού μηνυμάτων. Ο αλγόριθμος του συγχρονιστή-β από την άλλη πλευρά είναι αργός ως προς την εκτέλεση όλων των διαδικασιών αλλά υπερτερεί ως προς τον αριθμό των μηνυμάτων που είναι μικρός. Επομένως, ο αλγόριθμος του συγχρονιστή-γ συνδυάσει τα καλά στοιχεία των συγχρονιστών α και β και καταλήγει να είναι αποδοτικός τόσο από πλευράς χρόνου όσο και από πλευράς αριθμού μηνυμάτων που αποστέλλονται.

Ένας κόμβος μπορεί να προχωρήσει στον επόμενο παλμό εάν κάθε μήνυμα που έχει μεταδώσει εντός του συγκεκριμένου παλμού έχει παραδοθεί με επιτυχία (μέσω των γειτονικών κόμβων) στον τελικό προορισμό του. Ο έλεγχος αυτής της φάσης πραγματοποιείται από τους κοντινούς κόμβους μέσω της μετάδοσης ενός μηνύματος αποδοχής (ack). Όταν ένας κόμβος ανακαλύπτει ότι είναι ασφαλής, επικοινωνεί αυτές τις πληροφορίες σε όλους τους γύρω κόμβους του. Τώρα, με τη σειρά, κάθε γείτονας περνά στο επόμενο στάδιο αφού επιβεβαιώσει ότι όλοι οι άλλοι γείτονές του είναι ασφαλείς. Αυτό είναι το κριτήριο που πληρούν οι τρεις προτεινόμενοι αλγόριθμοι του Awerbuch (συγχρονιστής α, β και γ). Κάθε μέθοδος ειδοποιεί έναν κόμβο ότι οι γείτονές του είναι ασφαλείς με διαφορετικό τρόπο.

Ο Awerbuch ορίζει τους τρεις συγχρονιστές του εισάγοντας πρώτα την ιδέα της ασφάλειας. Ένας κόμβος του συστήματος θεωρείται ασφαλής όσον αφορά έναν παλμό (βήμα συγχρονισμένου αλγορίθμου) εάν κάθε μήνυμα συγχρονισμένου αλγορίθμου που παραδόθηκε από τον εν λόγω κόμβο κατά τη διάρκεια αυτού του παλμού έχει

φτάσει στον προβλεπόμενο παραλήπτη. Ένας κόμβος μπορεί να καθορίσει αν είναι ασφαλής αν υποθέσει ότι λαμβάνει επιβεβαίωση (acknowledgement) για κάθε μήνυμα που αποστέλλεται σε έναν γείτονα. Όταν όλες οι επικοινωνίες του κόμβου έχουν επιβεβαιωθεί, ο κόμβος είναι πλήρως ασφαλής. Έτσι, σύμφωνα με τις παραδοχές του μοντέλου δικτύου, κάθε κόμβος μαθαίνει ότι είναι ασφαλής σε ορισμένο χρονικό διάστημα μετά την εισαγωγή ενός νέου παλμού.

Συγχρονιστής-α

Η εκκίνηση του πρωτοκόλλου ξεκινάει με τον αλγόριθμο συγχρονιστή-α. Εκεί γίνεται η εκλογή αρχηγού στο κατανεμημένο σύστημα και έπειτα γίνεται η κατασκευή ενός γεννητικού δέντρου με ρίζα τον κόμβο που έχει τον ρόλο του αρχηγού. Η ρίζα δίνει το σήμα για την ενεργοποίηση της διαδικασίας. Όλοι οι κόμβοι – παιδιά της ρίζας στο δέντρο μόλις γνωρίσουν ότι οι ίδιοι είναι ασφαλείς το αναφέρουν στον γονέα τους αντίστοιχα. Με αυτό τον τρόπο υπάρχει πλήρη αποδοτικότητα μηνυμάτων, παρότι δεν καταναλώνεται πολύς χρόνος.

Θέτουμε σαν Σ τον αριθμό όλων των κόμβων σε ένα γράφο, και Α τον αριθμό του αντίστοιχου συνόλου των ακμών. Επιπλέον, γνωρίζουμε ότι ανά παλμό από κάθε ακμή περνούν σε διαφορετική κατεύθυνση ακριβώς δύο μηνύματα. Άρα ανά παλμό ρολογιού έχουμε 2Α μηνύματα, δηλαδή Ο(Α). Κάθε πλήρης γράφος με Σ κόμβους έχει το πολύ Σ(Σ-1)/2 ακμές, που σημαίνει $O(\Sigma^2)$ μηνύματα. Εδώ γνωρίζουμε ότι κάθε μήνυμα καθυστερεί 1 χρονική μονάδα, για τον λόγο ότι για κάθε παλμό έχουμε χρόνο Ο(1) για τα μηνύματα που μεταφέρονται μεταξύ γειτονικών κόμβων.

Συγχρονιστής-β

Ο επόμενος αλγόριθμος αφορά τον συγχρονιστή-β. Στο πλαίσιο αυτής της προσέγγισης, η προώθηση στον επόμενο παλμό ή το πόσο θα καθυστερήσει μέχρι να γίνει αυτή η προώθηση δεν πραγματοποιείται τοπικά σε κάθε κόμβο. Σε αυτή την περίπτωση όλοι οι κόμβοι συλλογικά μεταβαίνουν από παλμό σε παλμό όταν είναι αρχικά ασφαλής και έχουν ολοκληρώσει το προηγούμενο βήμα.

Αυτός ο συγχρονιστής χρειάζεται ένα βήμα έναρξης στο οποίο επιλέγεται ένας κόμβος-αρχηγός του δικτύου και σχηματίζεται ένα γενετικό δέντρο χρησιμοποιώντας τον κόμβο-αρχηγό του δικτύου ως ρίζα. Ο αρχηγός διατάσσει όλους τους κόμβους του δικτύου να ξεκινήσουν τον επόμενο παλμό διαδίδοντας ένα συγκεκριμένο σήμα στο δέντρο. Αφού ολοκληρωθεί η επεξεργασία ενός παλμού, ένας μηχανισμός που ονομάζεται *"convergecast"* χρησιμοποιείται για να πει στον αρχηγό ότι όλοι οι κόμβοι του συστήματος είναι ασφαλείς. Συγκεκριμένα, μόλις ένας κόμβος διαπιστώσει ότι είναι ασφαλής και ενημερωθεί ότι το ίδιο ισχύει και για όλα τα "παιδιά" του στο δέντρο, αναμεταδίδει αυτή την πληροφορία στον "γονέα" του στο δέντρο. Η πληροφορία αυτή ρέει από τα φύλλα του δέντρου προς τον αρχηγό στη βάση του δέντρου. Όταν τελικά ο αρχηγός ενημερωθεί ότι όλοι οι κόμβοι του συστήματος είναι ασφαλείς, διαδίδει και πάλι μέσω του δέντρου μια εντολή προς όλους τους κόμβους του συστήματος να συνεχίσουν με τον επόμενο παλμό.

Σύμφωνα με το παράδειγμα που αναπτύχθηκε παραπάνω το δέντρο που δημιουργήθηκε έχει Σ-1 ακμές. Επειδή έχουμε δύο μηνύματα σε κάθε ακμή κάθε παλμού, αυτό δείχνει ότι, στη χειρότερη περίπτωση, μεταφέρονται Ο(Σ) μηνύματα σε κάθε παλμό. Ο χρόνος καθορίζεται κυρίως από το ύψος του δέντρου. Δεδομένου ότι το ύψος του δέντρου δεν μπορεί να υπερβαίνει το Σ-1, εξετάζουμε στο χειρότερο σενάριο να έχουμε Ο(Σ) μονάδες χρόνου.

Συγχρονιστής-γ

Ο αλγόριθμος συγχρονιστής-γ είναι το αποτέλεσμα του συνδυασμού των δύο προηγούμενων. Εν συντομία, είναι μια εφαρμογή του αλγορίθμου α σε σύνολα κόμβων στα οποία εφαρμόζεται ο αλγόριθμος β σε κάθε μεμονωμένο κόμβο. Αρχικά ένα δάσος από γενετικά δέντρα δημιουργείται κατά τη διάρκεια της απαιτούμενης περιόδου εκκίνησης για τον συγχρονιστή γ. Η ομάδα των κόμβων στο σύνολό της χωρίζεται σε ένα σύνολο υποομάδων (συστάδες). Για κάθε συστάδα (cluster), επιλέγεται ένας αρχηγός. Αυτό στη συνέχεια οδηγεί στο σχηματισμό ενός γεννητικού δέντρου με τις ρίζες του στο επίπεδο του αρχηγού. Είναι απαραίτητο να επιλεχθεί μια προτιμώμενη σύνδεση, εάν υπάρχουν δύο κοντινά ξένα υποσύνολα. Καθώς εκτελείται η διαδικασία, καθένας από τους κόμβους στο άμεσο περιβάλλον ενός cluster καθίσταται ασφαλής και ως άμεση συνέπεια αυτού, εκκινείται ο επόμενος παλμός.

Παρακάτω παρουσιάζονται οι τύποι μηνυμάτων που ανταλλάσσονται:

- PULSE (ΠΑΛΜΟΣ): Ενεργοποίηση νέου παλμού στους κόμβους ενός cluster
- SAFE (ΑΣΦΑΛΗΣ): Μήνυμα όπου κάποιος κόμβος δηλώνει ότι είναι ασφαλής
- CLUSTER_SAFE (ΑΣΦΑΛΗΣ ΣΥΜΠΛΕΓΜΑ): Μήνυμα το οποίο δηλώνει ότι η συστάδα είναι ασφαλής. Γίνεται διάδοση σε όλη τη συστάδα καθώς και στις γειτονικές συστάδες μέσω των προτιμώμενων συνδέσεων.
- READY (ΕΤΟΙΜΟΣ): Μήνυμα το οποίο δηλώνει ότι ένας κόμβος είναι έτοιμος (ready) για τον επόμενο παλμό. Η διάδοση του συγκεκριμένου μηνύματος από τα φύλλα προς τον αρχηγό πραγματοποιείται μέσα σε μία συστάδα στην περίπτωση που ο κόμβος έχει λάβει τα αναγκαία μηνύματα READY από τα παιδιά του, καθώς και τα αναγκαία CLUSTER_SAFE από τις προτιμώμενες συνδέσεις, οι οποίες καταλήγουν σε αυτόν

Από την προηγούμενη αναφορά μας, θέτουμε σαν **Σ** το σύνολο των κόμβων του κατανεμημένου συστήματος, **Α** το σύνολο των ακμών-συνδέσεων, που είτε ανήκουν σε κάποιο δέντρο είτε θεωρούνται προτιμώμενες συνδέσεις, και **Υ** το μέγιστο ύψος από όλα τα δέντρα. Τα μηνύματα PULSE, READY, SAFE, και CLUSTER_SAFE μεταφέρονται σε κάθε παλμό μέσω των ακμών των δέντρων, μια φορά έκαστο. Επίσης, τα μηνύματα CLUSTER_SAFE και READY περνούν από μία φορά το καθένα, από κάθε προτιμώμενη σύνδεση. Έτσι καταλήγουμε σε ένα σύνολο με O(A) μηνύματα ανά παλμό ρολογιού. Επίσης, για κάθε σύμπλεγμα καταλήγουμε στο ότι O(Y) είναι ο χρόνος ενός παλμού, και O(Y) ο χρόνος διάδοσης των CLUSTER_SAFE μηνυμάτων,, το οποίο κατά ακολουθία σημαίνει ότι έχουμε συνολικό χρόνο O(Y).

7.3.3 Ο αλγόριθμος του LeLann

Ο LeLann ήταν ο πρώτος που παρουσίασε έναν αλγόριθμο εκλογής αρχηγού σε ένα κατανεμημένο σύστημα ονομασμένων κυκλικά διατεταγμένων επεξεργαστών (σε μια μονόδρομη τοπολογία δακτυλίου). Για την αντικατάσταση ενός χαμένου token κατά τη λειτουργία ενός δακτυλίου επεξεργαστών, ο χρήστης θα πρέπει να επικοινωνήσει με τον διαχειριστή του συστήματος. Κάθε μεμονωμένος αρχικοποιητής-επεξεργαστής στέλνει ένα μήνυμα στα υπόλοιπα συστήματα προσδιορίζοντας ποιος είναι ο αριθμός. Κάθε αρχικοποιητής-επεξεργαστής έχει μια ολοκληρωμένη λίστα που περιλαμβάνει τα αριθμητικά αναγνωριστικά που έχουν όλοι οι υπόλοιποι επεξεργαστές. Για να μάθουμε ποιος είναι ο εκλεγμένος νικητής, πρέπει να βρούμε τη μέγιστη τιμή στη λίστα του και αν αυτή η τιμή είναι ίδια με τον αριθμό του ίδιου του αρχικοποιητή, τότε αυτός είναι ο

νικητής. Υποτίθεται ότι κάθε κανάλι είναι FIFO και ότι κάθε επεξεργαστής-αρχικοποιητής πρέπει να δημιουργήσει το δικό του διακριτικό πριν λάβει ένα διακριτικό από οποιονδήποτε άλλο επεξεργαστή-αρχικοποιητή. Αυτός ο αλγόριθμος είναι αρκετά απλός στην εφαρμογή του με γραμμικό χρόνο για επικοινωνίες, αλλά απαιτεί μεγάλο αριθμό μεγάλου μεγέθους μηνυμάτων για να ταξιδέψει μεταξύ δύο κόμβων.

Παρακάτω παρουσιάζεται ο αλγόριθμος σε μορφή ψευδοκώδικα:

```
var Listl : set of L                    init{l};
        statel;
BEGIN
        If l is initiator then /* Εάν ο l(επεξεργαστής) είναι αρχικοποιητής */
                Begin
                        statel:=candidate; /* κατάσταση υποψήφιος */
                        send <tok,l> to Nextl; /* στέλνει την ταυτότητά του */
                        receive <tok,q>;
                        while p≠l do
                                begin /*τοποθετεί τα μηνύματα των άλλων*/
                                        Listl:=Listl ∪ {p}; /* στη λίστα του */
                                        send <tok,p> to Nextl; /* τα προωθεί */
                                        receive <tok,p>
                                end;
                        if l=max(Listl) then statel:=leader /*συνθήκη εκλογής*/
                        else statel:=lost
                end;
        else repeat receive <tok,p>; /*αν δεν είναι αρχικοποιητής*/
                send <tok,p> to Nextl; /*απλώς προωθεί τα μηνύματα*/
                if statel=sleep then statel:=lost
        until false
end
```

Κώδικας 7.2: Ο αλγόριθμος του LeLann

Το χαρακτηριστικό του αλγορίθμου LeLann (Κώδικας 7.2) είναι ότι αναφέρεται σε τοπολογίες δακτυλίου. Στον δακτύλιο έχουμε στην καλύτερη περίπτωση N διαφορετικά tokens με το καθένα να κάνει N βήματα. Οπότε έχουμε ως πολυπλοκότητα των μηνυμάτων O(N2). Ο πρώτος αρχικοποιητής στέλνει το δικό του token σε N-1 χρονικές μονάδες. Οποιοσδήποτε άλλος αρχικοποιητής έχει στείλει το δικό του μετά από N χρονικές μονάδες, όπως αντίστοιχα και για κάθε αρχικοποιητή, όπου μετά από N χρονικές μονάδες λαμβάνει το δικό του token. Τελικά, ο αλγόριθμος ολοκληρώνεται το πολύ σε 2N-1 χρονικές μονάδες.

7.3.4 Ο αλγόριθμος των Chang & Roberts

Ο αλγόριθμος των Chang & Roberts είναι ταχύτερος από τον αλγόριθμο LeLann, καθώς απαιτεί, λιγότερα μηνύματα κατά μέσο όρο για αποστολή σε γραμμικό χρόνο. Ο αριθμός των μηνυμάτων στη χειρότερη περίπτωση του αλγόριθμου LeLann εξακολουθεί να είναι ισοδύναμος με τον αριθμό των μηνυμάτων στην καλύτερη περίπτωση του αλγόριθμου LeLann. Το πρωτόκολλο δικτύου έχει μια σειρά από πλεονεκτήματα, συμπεριλαμβανομένης της επεκτασιμότητας σε άλλες τοπολογίες, της εύκολης υλοποίησης, καθώς και πολύ καλή απόδοση υπό συγκεκριμένες συνθήκες χρονισμού δικτύου. Ο αλγόριθμος χαρακτηρίζει την άγνοια του συνολικού αριθμού των επεξεργαστών που συμμετέχουν στην εκλογή, την κίνηση των μηνυμάτων προς μία μόνο κατεύθυνση και τη δυνατότητα μην υπάρχει ταυτόχρονη εκκίνηση από όλους τους επεξεργαστές. Η μη ταυτόχρονη εκκίνηση είναι πιο πιθανό να οδηγήσει σε καλύτερα αποτελέσματα από εκείνα του σύγχρονου ξεκινήματος.

Το βασικό χαρακτηριστικό αυτού του αλγορίθμου είναι αρχικά να μαντέψει και έπειτα να επιλέξει ποια μηνύματα θα καταστρέψει από τους επεξεργαστές που δεν έχουν πιθανότητα να εκλεγούν αρχηγοί. Για να μην έχει πιθανότητες ένας επεξεργαστής να βρεθεί νικητής θα πρέπει να βρεθεί ένας άλλος με αριθμό μεγαλύτερο από τον δικό του. Γενικότερα, σε αυτήν την παραλλαγή εκλέγεται αρχηγός ο επεξεργαστής με το μεγαλύτερο νούμερο.

Πιο συγκεκριμένα, ο αλγόριθμος αρχικά έχει ως δεδομένο ότι κάθε επεξεργαστής γνωρίζει τον αριθμό του. Το επόμενο βήμα είναι να καταλάβει ο επεξεργαστής ότι έχει μπει σε διαδικασία εκλογής αρχηγού, κάτι που φαίνεται με δύο τρόπους. Ο πρώτος είναι να ειδοποιηθεί από κάποιον άλλον γειτονικό κόμβο ή να το καταλάβει από μόνος του. Όλες αυτές οι ειδοποιήσεις υλοποιούνται με την αποστολή του αριθμού του κάθε επεξεργαστή από τον ίδιο τον επεξεργαστή στον αριστερό γείτονά του.

Το επόμενο βήμα αφορά τον επεξεργαστή που παίρνει το μήνυμα, ο οποίος το συγκρίνει με τον δικό του αριθμό. Στο επόμενο στάδιο συγκρίνει τον αριθμό του και αν είναι μικρότερος, τότε πιθανόν το μήνυμα που έλαβε να είναι το μήνυμα του νικητή, και για αυτό το στέλνει προς τον αριστερό κόμβο. Αντιθέτως, εάν το μήνυμα περιέχει μικρότερο αριθμό από τον δικό του, τότε αποκλείεται η περίπτωση να είναι αυτός ο αριθμός του αρχηγού με αποτέλεσμα το μήνυμα να αγνοείται ή να διαγράφεται. Στο τελικό στάδιο, αν κατά τη σύγκριση του δικού του αριθμού με τον αριθμό του μηνύματος φανεί ότι είναι ίδιοι, τότε καταλαβαίνει πως ο δικός του αριθμός έχει περάσει από όλους τους κόμβου και ξαναέφτασε σε αυτόν. Που σημαίνει ότι αυτός είναι ο αρχηγός. Σε αυτό το σημείο η διαδικασία τερματίζεται.

Ο αλγόριθμος έχει δημιουργηθεί με σκοπό την εύρεση του επεξεργαστή με τον μέγιστο αριθμό. Η τοπολογία δακτυλίου και η μοναδική δρομολόγηση των μηνυμάτων καθιστούν δύσκολο για ένα μήνυμα να ταξιδέψει μέσω του δικτύου χωρίς να περάσει από όλους τους επεξεργαστές. Ο αλγόριθμος έχει σχεδιαστεί έτσι ώστε το μήνυμα που έχει τον μεγαλύτερο αριθμό να ολοκληρώσει έναν ολοκληρωμένο κύκλο χωρίς να αποκοπεί από κανέναν από τους ενδιάμεσους επεξεργαστές. Ένας επεξεργαστής-κόμβος επιλέγεται να είναι ο μοναδικός επικεφαλής της εκλογικής διαδικασίας με βάση τον αριθμό του. Σε ένα κατανεμημένο σύστημα κάθε επεξεργαστής μπορεί να μοντελοποιηθεί ως ένας αυτοματοποιημένος κόμβος που, ανάλογα με την τρέχουσα κατάστασή του και το μήνυμα που λαμβάνει, θα μεταβεί σε μια νέα κατάσταση. Ένα στιγμιότυπο της κατάστασης των επεξεργαστών του

συστήματος μπορεί να θεωρηθεί ως ένα διάνυσμα που αποτελείται από όλες τις τρέχουσες καταστάσεις του επεξεργαστή.

Παρακάτω αναφέρονται κάποιοι τεχνικοί κανόνες που αφορούν λεπτομέρειες για την λειτουργία του αλγορίθμου.

- Ένας ανενεργός (sleeping) κόμβος
    - Έχει τη δυνατότητα να ξυπνάει μόνος του, να ενεργοποιείται και να στέλνει ένα μήνυμα κειμένου με τον αριθμό του στα αριστερά.
    - Εάν ένας κόμβος ενεργοποιηθεί από το μήνυμα ενός άλλου κόμβου, θα περάσει το μήνυμα στην περίπτωση που ο αριθμός του είναι μικρότερος από τον αριθμό του μηνύματος που τον ξύπνησε. Διαφορετικά, θα αγνοήσει το μήνυμα αφύπνισης και θα στείλει το δικό του μήνυμα.
- Ένας ενεργός κόμβος (awake node)
    - Θα αγνοήσει το μήνυμα εάν σταλεί με μικρότερο αριθμό από αυτόν που έχει τώρα.
    - Όταν λαμβάνει ένα μήνυμα με έναν αριθμό που είναι μεγαλύτερος από τον δικό του, περνάει τις πληροφορίες.
    - Αν λάβει μήνυμα με τον ίδιο αριθμό με το δικό του, τότε οι εκλογές έχουν τελειώσει και ανακηρύσσεται νικητής.

Ο αλγόριθμος των Chang & Roberts σε μορφή κώδικα είναι ο εξής:

Κόμβος k : όταν δέχεται ένα μήνυμα r :

Κατάσταση κόμβου = ανενεργός

BEGIN

IF k > r THEN send k

ELSE IF k < r THEN send r

κατάσταση := ενεργός

END

Κατάσταση κόμβου = ενεργός

BEGIN

IF k > r THEN skip

ELSE IF k < r THEN send r

ELSE IF k = r THEN κατάσταση := ανακηρύσσεται νικητής

END

Κώδικας 7.3: Ο αλγόριθμος των Chang & Roberts

Αφού ολοκληρωθεί ο αλγόριθμος, ο επεξεργαστής που επιλέχθηκε να τερματίσει τη διαδικασία στέλνει ένα μήνυμα στους άλλους επεξεργαστές δικτύου ενημερώνοντάς τους ότι η διαδικασία έχει τελειώσει.

Ο αλγόριθμος των Chang & Roberts δίνει λύση στο πρόβλημα εκλογής αρχηγού σε τοπολογία δακτυλίου χρησιμοποιώντας Θ(N2) μηνύματα για την περίπτωση που ο

μέγιστος αριθμός περάσει από όλους τους κόμβους του δακτυλίου. Χρησιμοποιούνται στη χειρότερη περίπτωση N tokens, όπου κάθε token διαδίδεται το πολύ σε N βήματα. Οπότε, συνολικά έχουμε O(N2) μηνύματα. Ας υποθέσουμε την περίπτωση τοποθέτησης των κόμβων με αύξουσα σειρά πάνω στο δακτύλιο, που σημαίνει ότι όλα τα tokens «κόβονται» από τον κόμβο 0, εκτός φυσικά του πρώτου κόμβου. Τότε, κάθε token i προωθείται κατά N-i βήματα. Οπότε, το σύνολο των μηνυμάτων είναι (1/2)N(N+1). Το αποτέλεσμα είναι O(N2) μηνύματα. Ο συγκεκριμένος αλγόριθμος σε μία μέση περίπτωση που θεωρεί ως αρχικοποιητές όλους τους κόμβους απαιτεί O(NlogN) μηνύματα.

Βιβλιογραφικές αναφορές

[1] Andrew S. Tanenbaum, Maarten Van Steen (2006), Κατανεμημένα Συστήματα: Αρχές και Υποδείγματα, Έκδοση: 1η/2006, Εκδόσεις Κλειδάριθμος.

[2] Coulouris, J. Dollimore, T. Kindberg, G. Blair (2020), Κατανεμημένα Συστήματα, Έκδοση: 2η, Da Vinci Μ.Ε.Π.Ε.

[3] Κάβουρας Ι.Κ., Μήλης Ι.Ζ., Ρουκουνάκη Α.Α., Ξηλωμένος Γ.Β. (2011), Κατανεμημένα συστήματα σε Java, 3η έκδοση, Εκδόσεις Κλειδάριθμος ΕΠΕ.

[4] David Culler, J.P. Singh, and Anoop Gupta. Parallel Computer Architecture: A Hardware/Software Approach. Morgan Kaufmann. ISBN-10: 1558603433, 1998.

[5] Παπαδάκης Στ., Διαμαντάρας Κ., Προγραμματισμός και Αρχιτεκτονική Συστημάτων Παράλληλης Επεξεργασίας, Εκδόσεις Κλειδάριθμος, 2012.

[6] Δημακόπουλος Β., 2015, 'Παράλληλα Συστήματα και Προγραμματισμός', [ηλεκτρ. βιβλ.] Αθήνα: Σύνδεσμος Ελληνικών Ακαδημαϊκών Βιβλιοθηκών. Διαθέσιμο στο: https://repository.kallipos.gr/bitstream/11419/3209/4/book.pdf .

[7] Leslie Lamport and P. M. Melliar-Smith. 1985. Synchronizing clocks in the presence of faults. J. ACM 32, 1 (Jan. 1985), 52–78. https://doi.org/10.1145/2455.2457

[8] Σημειώσεις Μαθήματος "Αλγόριθμοι Κατανεμημένων συστημάτων" 2010 ΤΕΙ Λαμίας - Κωνσταντίνος Αντωνής

[9] Introduction to Distributed Algorithms, Gerard Tel, Cambridge University Press, 1994

[10] Distributed Algorithms, Nancy Lynch, Morgan Kaufman, 1996

Κριτήρια αξιολόγησης

Ερώτηση 1

Ο αλγόριθμος των Chang & Roberts είναι ταχύτερος από τον αλγόριθμο του LeLann, καθώς απαιτεί, κατά μέσο όρο, περισσότερα μηνύματα για αποστολή σε γραμμικό χρόνο.

α. Σωστό

β. Λάθος

Ερώτηση 2

Ο αλγόριθμος Chang & Roberts λύνει το πρόβλημα εκλογής αρχηγού σε τοπολογία δακτυλίου χρησιμοποιώντας Θ(N2) μηνύματα στη περίπτωση που … περάσει από όλους τους κόμβους του δακτυλίου

α. ο αριθμός του πλήθους των γειτονικών κόμβων του αρχηγού

β. ο ελάχιστος αριθμός.

γ. ο μέγιστος αριθμός

Ερώτηση 3

Ο αμοιβαίος αποκλεισμός απαιτεί:

α. πολλές διεργασίες να εκτελούν λειτουργίες σε ιδιωτικούς πόρους

β. μία μόνο διεργασία να εκτελεί λειτουργίες σε ιδιωτικούς πόρους

γ. μία μόνο διεργασία να εκτελεί λειτουργίες σε κοινούς πόρους

δ. πολλές διεργασίες να εκτελούν λειτουργίες σε κοινούς πόρους

Ερώτηση 4

Οι σημαφόροι είναι:

α. ακέραιες μεταβλητές, που μοιράζονται μεταξύ πολλών διεργασιών

β. δεκαδικές μεταβλητές, που μοιράζονται μεταξύ πολλών διεργασιών

γ. ακέραιες μεταβλητές, που ελέγχονται από την διεργασία τον κεντρικό κόμβο

δ. δυαδικές μεταβλητές, που μοιράζονται μεταξύ πολλών διεργασιών

Ερώτηση 5

Στον αλγόριθμο των Chang & Roberts ένας ανενεργός (sleeping) κόμβος μπορεί να αφυπνιστεί αυτόματα, από μόνος του, να ενεργοποιηθεί και να στείλει μήνυμα με τον αριθμό του προς τα δεξιά

α. Σωστό

β. Λάθος

Ερώτηση 6

Ο αμοιβαίος αποκλεισμός απαιτεί μία μόνο διεργασία να εκτελεί λειτουργίες σε κοινούς πόρους.

α. Σωστό

β. Λάθος

Ερώτηση 7

Στον αλγόριθμο διάδοσης του δένδρου για να υπάρξει πρόοδος στον αλγόριθμο στην περίπτωση που μερικές μόνο από τις διεργασίες είναι αρχικοποιητές, προστίθεται μία φάση

α. αναδιαμόρφωσης φύλλων

β. προσωρινής απενεργοποίησης (sleeping)

γ. αφύπνισης (wake up)

δ. κανένα από τα παραπάνω

Ερώτηση 8

Στον αλγόριθμο διάδοσης του δένδρου εάν μία διεργασία έχει λάβει ένα μήνυμα από κάθε γείτονά της εκτός ενός, η διεργασία δεν στέλνει ένα μήνυμα προς τον υπολειπόμενο γείτονα

α. Σωστό

β. Λάθος

Ερώτηση 9

Στον αμοιβαίο αποκλεισμό, όσον αφορά τα κατανεμημένα συστήματα:

α. υπάρχει η έννοια την κοινής μνήμης και η έννοια του κοινού φυσικού ρολογιού

β. δεν υπάρχει η έννοια την κοινής μνήμης, αλλά υπάρχει η έννοια του κοινού φυσικού ρολογιού

γ. υπάρχει η έννοια την κοινής μνήμης, δεν υπάρχει η έννοια του κοινού φυσικού ρολογιού

δ. δεν υπάρχει η έννοια την κοινής μνήμης, ούτε του κοινού φυσικού ρολογιού

Ερώτηση 10

Ο αλγόριθμος Raymond βασίζεται σε διεργασίες που οργανώνονται σε ένα λογικό γεννητικό δένδρο (logical spanning tree), κάτι το οποίο μειώνει τον αριθμό των μηνυμάτων που ανταλλάσσονται σε O(N)

α. Σωστό

β. Λάθος

Ερώτηση 11

Η απαίτηση/απαιτήσεις αλγορίθμων αμοιβαίου αποκλεισμού είναι:

α. Να μην υπάρχει “αδιέξοδος” - No Deadlock

β. Να μην υπάρχει “λιμοκτονία” - No Starvation

γ. Να υπάρχει “δικαιοσύνη” - Fairness

δ. Να υπάρχει ανοχή σε σφάλματα - Fault Tolerance

ε. Όλα τα παραπάνω

Ερώτηση 12

Ποια πρόταση είναι αληθής όσον αφορά τους αλγορίθμους αμοιβαίου αποκλεισμού;

α. Οι κοινόχρηστες μεταβλητές μπορεί να χρησιμοποιηθούν για την εφαρμογή αμοιβαίου αποκλεισμού σε κατανεμημένα συστήματα, διότι οι κατανεμημένες διεργασίες δεν “βλέπουν” κάποια κοινή μνήμη.

β. Οι κοινόχρηστες μεταβλητές ή ένας τοπικός πυρήνας δεν μπορεί να χρησιμοποιηθούν για την εφαρμογή αμοιβαίου αποκλεισμού σε κατανεμημένα συστήματα, διότι οι κατανεμημένες διεργασίες δεν “βλέπουν” κάποια κοινή μνήμη.

γ. Οι κοινόχρηστες μεταβλητές ή ένας τοπικός πυρήνας δεν μπορεί να χρησιμοποιηθούν για την εφαρμογή αμοιβαίου αποκλεισμού σε κατανεμημένα συστήματα, διότι οι κατανεμημένες διεργασίες “βλέπουν” σίγουρα κάποια κοινή μνήμη.

δ. Οι κοινόχρηστες μεταβλητές ή ένας τοπικός πυρήνας μπορεί να χρησιμοποιηθούν για την εφαρμογή αμοιβαίου αποκλεισμού σε κατανεμημένα συστήματα, διότι οι κατανεμημένες διεργασίες “βλέπουν” κάποια κοινή μνήμη.

Ερώτηση 13

Ο αλγόριθμος που βασίζεται σε διακριτικά (Token Based):

α. Πολλά διακριτικά μοιράζεται σε όλες τις κρίσιμες περιοχές.

β. Εάν μία κρίσιμη περιοχή διαθέτει το μοναδικό αυτό διακριτικό, επιτρέπεται να εισέλθει στο κρίσιμο τμήμα της.

γ. Εάν μία κρίσιμη περιοχή διαθέτει το μοναδικό αυτό διακριτικό, δεν επιτρέπεται να εισέλθει στο κρίσιμο τμήμα της.

δ. Εάν μία κρίσιμη περιοχή διαθέτει πολλαπλά διακριτικά, επιτρέπεται να εισέλθει στο κρίσιμο τμήμα της.

Ερώτηση 14

Στους αλγόριθμους εκλογής αρχηγού(παραπάνω από μία σωστές απαντήσεις) :

α. Οι κρίσιμες περιοχές χωρίζονται σε εκλεγμένες και μη εκλεγμένες.

β. Όταν ένας κόμβος εισέρχεται σε μία οποιαδήποτε κατάσταση, πάντα παραμένει σε αυτή την κατάσταση.

γ. Όταν ένας κόμβος εισέρχεται σε μία οποιαδήποτε κατάσταση, μπορεί να μεταβεί σε άλλη κατάσταση.

δ. Η προϋπόθεση ασφαλείας για την εκλογή αρχηγού απαιτεί ότι όλοι οι κόμβοι μπορούν να εισέλθουν στην εκλεγμένη κατάσταση (να εκλεγούν αρχηγοί)

ε. Κάθε αλγόριθμος εκλογής αρχηγού πρέπει να ικανοποιεί τις συνθήκες ασφάλειας και λειτουργικότητας για να είναι δυνατή η εκτέλεσή του.

Ερώτηση 15

Ο αλγόριθμος του Maekawa είναι:

α. μια προσέγγιση βασισμένη στην απαρτία (Quorum based)

β. μια προσέγγιση βασισμένη σε διακριτικά (Token Based)

γ. μια προσέγγιση που δεν είναι βασισμένη σε διακριτικά (Non-Token Based)

δ. μια προσέγγιση βασισμένη σε αλγορίθμους εκλογής αρχηγού

Ερώτηση 16

Η προσέγγιση που δεν βασίζεται σε διακριτικά (Non-Token Based)

α. Όταν μία κρίσιμη περιοχή κάνει αίτημα για κρίσιμη ενότητα, δεν λαμβάνει καμία χρονική σήμανση

β. δεν απαιτεί την ανταλλαγή δύο ή περισσότερων διαδοχικών μηνυμάτων μεταξύ των κρίσιμων περιοχών

γ. δεν χρησιμοποιεί χρονικές σημάνσεις (timestamps)

δ. απαιτεί την ανταλλαγή δύο ή περισσότερων διαδοχικών μηνυμάτων μεταξύ των κρίσιμων περιοχών

Ερώτηση 17

Ο ρόλος της σκυτάλης στον αλγόριθμο Raymond είναι να ξεκινάει από τη ρίζα του δέντρου, έπειτα να περνάει από τις ακμές του δέντρου στο μονοπάτι με σκοπό να βρει τη διεργασία που έκανε το αίτημα

α. Σωστό

β. Λάθος

Ερώτηση 18

Κατά τον αλγόριθμο των Ricart & Agrawala στην περίπτωση που κάποια διεργασία εισέρχεται στην κρίσιμη περιοχή στέλνει μήνυμα στις υπόλοιπες διεργασίες.

όταν κάποια διεργασία παραλήπτης παραλαμβάνει τέτοιο μήνυμα και δεν θέλει να εισέλθει στην κρίσιμη περιοχή και, φυσικά, δεν βρίσκεται σε αυτή απαντά

α. θετικά

β. αρνητικά

γ. μπαίνει σε ανενεργή κατάσταση

δ. συγκρίνει τις χρονοσφραγίδες και απαντά αναλόγως

Ερώτηση 19

Μια φάση αφύπνισης (wakeup phase) εισάγεται στον αλγόριθμο Δένδρου εκλογής αρχηγού μόνο σε περίπτωση

α. που υπάρχουσα ρίζα σταματήσει να είναι αρχηγός

β. που η ρίζα είναι αρχικοποιητής

γ. που μία διεργασία είναι αρχικοποιητής

δ. που μέρος των διεργασιών είναι αρχικοποιητές

Ερώτηση 20

Ο αλγόριθμος του συγχρονιστή-α που ανήκει στην ομάδα αλγορίθμων του Awerbuch

α. μπορεί να προχωρήσει στον επόμενο παλμό εάν έστω ένα μήνυμα που έχει μεταδώσει εντός του συγκεκριμένου παλμού έχει παραδοθεί με επιτυχία

β. είναι αργός ως προς την εκτέλεση όλων των διαδικασιών αλλά υπερτερεί ως προς τον αριθμό των μηνυμάτων που είναι μικρός

γ. είναι αρκετά γρήγορος ως προς την υλοποίησή του, με μειονέκτημα την αποστολή μεγάλου αριθμού μηνυμάτων

δ. είναι αποδοτικός τόσο από πλευράς χρόνου όσο και από πλευράς αριθμού μηνυμάτων που αποστέλλονται.

Ερώτηση 21

Ο αλγόριθμος του συγχρονιστή-β που ανήκει στην ομάδα αλγορίθμων του Awerbuch

α. μπορεί να προχωρήσει στον επόμενο παλμό εάν έστω ένα μήνυμα που έχει μεταδώσει εντός του συγκεκριμένου παλμού έχει παραδοθεί με επιτυχία

β. είναι αργός ως προς την εκτέλεση όλων των διαδικασιών αλλά υπερτερεί ως προς τον αριθμό των μηνυμάτων που είναι μικρός

γ. είναι αρκετά γρήγορος ως προς την υλοποίησή του, με μειονέκτημα την αποστολή μεγάλου αριθμού μηνυμάτων

δ. είναι αποδοτικός τόσο από πλευράς χρόνου όσο και από πλευράς αριθμού μηνυμάτων που αποστέλλονται.

Ερώτηση 22

Ο αλγόριθμος του συγχρονιστή-γ που ανήκει στην ομάδα αλγορίθμων του Awerbuch

α. μπορεί να προχωρήσει στον επόμενο παλμό εάν έστω ένα μήνυμα που έχει μεταδώσει εντός του συγκεκριμένου παλμού έχει παραδοθεί με επιτυχία

β. είναι αργός ως προς την εκτέλεση όλων των διαδικασιών αλλά υπερτερεί ως προς τον αριθμό των μηνυμάτων που είναι μικρός

γ. είναι αρκετά γρήγορος ως προς την υλοποίησή του, με μειονέκτημα την αποστολή μεγάλου αριθμού μηνυμάτων

δ. είναι αποδοτικός τόσο από πλευράς χρόνου όσο και από πλευράς αριθμού μηνυμάτων που αποστέλλονται.

Ερώτηση 23

Το χαρακτηριστικό του αλγορίθμου LeLann είναι ότι αναφέρεται σε τοπολογίες

α. Πλέγματος (mesh)

β. Δένδρων

γ. Δακτυλίου

δ. Όλα τα παραπάνω.