ΚΑΤΑΝΕΜΗΜΕΝΟΙ ΑΛΓΟΡΙΘΜΟΙ

ΜΑΡΙΟΣ ΜΑΥΡΟΝΙΚΟΛΑΣ

Τμήμα Πληροφορικής
Πανεπιστήμιο Κύπρου
Λευκωσία, Κύπρος

Σεπτέμβριος 1999

ΚΑΤΑΝΕΜΗΜΕΝΑ ΣΥΣΤΗΜΑΤΑ

ΜΕΡΟΣ ΤΗΣ ΚΑΘΗΜΕΡΙΝΗΣ ΖΩΗΣ ΜΑΣ

- Για να μοιραζόμαστε πόρους
- Για να επικοινωνούμε
- Για να πετυχαίνουμε καλύτερη επίδοση στους υπολογισμούς μας όσο αφορά
  - ταχύτητα
  - ανοχή σε σφάλματα

ΧΑΡΑΚΤΗΡΙΖΟΝΤΑΙ ΑΠΟ

- ανεξάρτητες δραστηριότητες (συνδρομικότητα)
- χαλαρό παραλληλισμό (ετερογένεια)
- εγγενή αβεβαιότητα

Η ΑΒΕΒΑΙΟΤΗΤΑ ΠΗΓΑΖΕΙ ΑΠΟ

- διαφορετικές ταχύτητες των επεξεργαστών
- διαφορετικές καθυστερήσεις στην επικοινωνία
- (μερικά) σφάλματα
- πολλαπλές ροές εισόδων και συμπεριφορά αλληλεπίδρασης

ΣΥΛΛΟΓΙΣΤΙΚΗ ΚΑΤΑΝΕΜΗΜΕΝΩΝ ΣΥΣΤΗΜΑΤΩΝ

Η αβεβαιότητα δυσχεραίνει τη συλλογιστική για τα κατανεμημένα συστήματα, αν θέλουμε να είμαστε βέβαιοι ότι δουλεύουν σωστά.
Ξεπερνάμε αυτή τη δυσκολία ακολουθώντας το θεωρητικό υπόδειγμα:

- Προσδιόρισε τα θεμελιώδη προβλήματα στο κατάλληλο επίπεδο αφαιρετικότητας.
- Όρισε με ακρίβεια τα προβλήματα αυτά.
- Σχεδίασε αλγόριθμους για να επιλύσεις αυτά τα προβλήματα.
- Απόδειξε την ορθότητα των αλγορίθμων.
- Ανάλυσε την πολυπλοκότητα των αλγορίθμων (π.χ., χρόνος, μνήμη, μηνύματα).
- Απόδειξε αποτελέσματα αδυνάτου και κάτω φράγματα.

Δυνατά κέρδη:

- Η προσεκτική διατύπωση προδιαγραφών αποσαφηνίζει τις προθέσεις.
- Αυξάνει η εμπιστοσύνη μας στην ορθότητα.
- Το κατάλληλο επίπεδο αφαιρετικότητας είναι χρήσιμο σε πολλαπλές περιπτώσεις.
- Μαθαίνουμε ποιοί είναι οι αναπόφευκτοι περιορισμοί.

ΠΕΡΙΟΧΕΣ ΕΦΑΡΜΟΓΩΝ

Μερικές περιοχές εφαρμογών έχουν συνεισφέρει κλασσικά προβλήματα στο πεδίο του κατανεμημένου υπολογισμού:

- λειτουργικά συστήματα.
- κατανεμημένες βάσεις δεδομένων.
- συστήματα πραγματικού χρόνου.
- δίκτυα επικοινωνιών
- αρχιτεκτονικές πολυεπεξεργαστών

ΕΠΙΣΚΟΠΗΣΗ ΜΑΘΗΜΑΤΟΣ – ΜΕΡΟΣ Ι (ΘΕΜΕΛΙΩΔΗ)

Θα εισάγουμε τα δύο βασικά μοντέλα επικοινωνίας:

- ανταλλαγή μηνυμάτων
- κοινόχρηστη μνήμη

και τα δύο βασικά μοντέλα χρονισμού:

- σύγχρονο
- ασύγχρονο

και θα μελετήσουμε (σχεδόν) όλους τους μεταξύ τους συνδυασμούς.

ΜΕΡΟΣ Ι (ΣΥΝΕΧΕΙΑ)

Θα καλύψουμε τα κανονικά προβλήματα και ζητήματα:

- αλγόριθμοι πάνω σε γράφους
- εκλογή προέδρου
- αμοιβαίος αποκλεισμός
- συμφωνία ανεκτική σε σφάλματα
- χρόνος και συγχρονισμός

ΕΠΙΣΚΟΠΗΣΗ ΜΑΘΗΜΑΤΟΣ – ΜΕΡΟΣ ΙΙ (ΠΡΟΧΩΡΗΜΕΝΑ ΘΕΜΑΤΑ)

Θα εμβαθύνουμε περισσότερο σε ορισμένες απο τις κατευθύνσεις που θα έχουμε δεί:

- πιθανοποίηση και πιθανοτικοί αλγόριθμοι
- κατανεμημένη κοινόχρηστη μνήμη (συνθήκες συνέπειας)
- ημισυγχρονισμός και μερικός χρονισμός
- κατανεμημένη λήψη αποφάσεων
- μετρητές και δίκτυα μέτρησης
- επικοινωνιακά προβλήματα
  - διαχείριση διασυνδέσεων
  - έλεγχος ροής
- τυπικές μέθοδοι για απόδειξη ορθότητας κατανεμημένων αλγορίθμων

ΣΧΕΣΗ ΤΗΣ ΘΕΩΡΙΑΣ ΠΡΟΣ ΤΗΝ ΠΡΑΞΗ

- χρονικά καταμερισμένο λειτουργικό σύστημα: ζητήματα σχετικά προς την (ιδεατή) συνδρομικότητα των διεργασιών, οπως, π.χ.,
  - αμοιβαίος αποκλεισμός
  - αδιέξοδο
- πολυεπεξεργαστές:
  - αν υπάρχει κοινό κεντρικό ρολόι, τότε το σύγχρονο μοντέλο ταιριάζει, αλλοιώς το ασύγχρονο
- χαλαρά συνδεδεμένα επικοινωνιακά δίκτυα, όπως το Internet: το ασύγχρονο μοντέλο με ανταλλαγή μηνυμάτων ταιριάζει

Μοντέλα σφαλμάτων:

- κατάρρευση επεξεργαστών
- Βυζαντινά (αυθαίρετα) σφάλματα: συντηρητική υπόθεση, είναι κατάλληλη όταν ο τρόπος με τον οποίο συμβαίνουν τα σφάλματα δεν είναι γνωστός ή μπορεί να είναι κακοπροαίρετος

ΜΟΝΤΕΛΟ ΑΝΤΑΛΛΑΓΗΣ ΜΗΝΥΜΑΤΩΝ

- οι επεξεργαστές είναι $p_0, p_1, \ldots, p_{n-1}$ και βρίσκονται τοποθετημένοι στους κόμβους του γράφου
- υπάρχουν σημείο-προς-σημείο, διπλής κατευθυνσης, κανάλια επικοινωνίας (μη κατευθυνόμενες ακμές του γράφου)
- κάθε επεξεργαστής χρωματίζει τα προσπίπτοντα σ'αυτόν κανάλια με τους ακέραιους $1, 2, 3, \ldots$ (δυνατό να μη γνωρίζει ποιος επεξεργαστής βρίσκεται στο αντίθετο άκρο)

ΕΠΕΞΕΡΓΑΣΤΕΣ, ΚΑΝΑΛΙΑ ΚΑΙ ΔΙΑΤΑΞΕΙΣ

- Ο επεξεργαστής είναι μια μηχανή καταστάσεων (δυνατό να περιλαμβάνει άπειρο αριθμό καταστάσεων).
- Κάθε κατάσταση είναι μια συλλογή από μεταβλητές κατάστασης.
- Δυνατό να υπάρχουν μεταβλητές κατάστασης τις οποίες ο επεξεργαστής δεν βλέπει – μη προσπελάσιμες μεταβλητές κατάστασης.

Διάταξη: το διάνυσμα των καταστάσεων των επεξεργαστών. Συλλαμβάνει την τρέχουσα εικόνα ολοκλήρου του συστήματος.

ΓΕΓΟΝΟΤΑ

Υπάρχουν δύο τύποι γεγονότων:

1. Παραλαβή μηνύματος: ένα μήνυμα μετακινείται από τον προς-τα-έξω προσωρινό καταχωρητή μηνυμάτων του αποστολέα στον προς-τα-έσω προσωρινό καταχωρητή μηνυμάτων του παραλήπτη. Το μήνυμα θα τύχει επεξεργασίας στο επόμενο υπολογιστικό βήμα του παραλήπτη.
2. Υπολογισμός: ένα υπολογιστικό βήμα από κάποιο επεξεργαστή. Συμβολίζει μια μετάβαση της μηχανής καταστάσεων του επεξεργαστή, στην οποία όλα τα εισερχόμενα μηνύματα τυγχάνουν επεξεργασίας.

Ένα υπολογιστικό βήμα μετασχηματίζει μια αρχική κατάσταση (το προσπελάσιμο μέρος της) σε μια τελική κατάσταση όπου ο προς-τα-έσω προσωρινός καταχωρητής μηνυμάτων του επεξεργαστή είναι κενός. Επίσης, προσδιορίζει ποια μηνύματα θα στείλει ο επεξεργαστής.

ΑΣΥΓΧΡΟΝΗ ΕΚΤΕΛΕΣΗ

Εκτέλεση:
διάταξη, γεγονός, διάταξη, γεγονός, ..., διάταξη, γεγονός, διάταξη, ...

- στην αρχική διάταξη, κάθε επεξεργαστής βρίσκεται στην αρχική του διάταξη (η οποία προσδιορίζεται από τον αλγόριθμό του) και όλοι οι προς-τα-έσω προσωρινοί καταχωρητές μηνυμάτων είναι κενοί.
- για κάθε συνεχόμενη τριάδα (διάταξη, γεγονός, διάταξη), η νέα διάταξη είναι ακριβώς η ίδια με την παλιά εκτός από τις εξής διαφορές:
  - αν το γεγονός είναι παραλαβή μηνύματος, τότε το μήνυμα που παραλαμβάνεται μετακινείται από τον προς-τα-έξω προσωρινό καταχωρητή μηνυμάτων του αποστολέα στον προς-τα-έσω προσωρινό καταχωρητή μηνυμάτων του παραλήπτη.
  - αν το γεγονός είναι υπολογισμός, τότε η κατάσταση του επεξεργαστή που εκτελεί το βημα (μαζί με τον προς-τα-έξω προσωρινό καταχωρητή μηνυμάτων του) αλλάζει σύμφωνα με τη συνάρτηση μετάβασής του.

ΝΟΜΙΜΕΣ ΕΚΤΕΛΕΣΕΙΣ

Ο ορισμός της εκτέλεσης προσδιορίζει συνθήκες ασφαλείας: συνθήκες που πρέπει να ισχύουν σε κάθε πεπερασμένο πρόθεμα – τίποτα κακό δεν συμβαίνει.
Πολλές φορές θέλουμε να επιβάλουμε συνθήκες ζωτικότητας: πράγματα που πρέπει να συμβούν κάποιο αριθμό από φορες, ενδεχομένως άπειρα πολλές φορές – τελικά, κάτι καλό συμβαίνει.
Νόμιμες εκτελέσεις: ικανοποιούν επιπρόσθετα τις αρμόζουσες συνθήκες ζωτικότητας – κάνουν το σωστό πράγμα.

Για το ασύγχρονο μοντέλο, νόμιμη εκτέλεση σημαίνει:

- κάθε μήνυμα σε ένα προς-τα-έξω προσωρινό καταχωρητή παραλαμβάνεται τελικά
- κάθε επεξεργαστής εκτελεί ένα άπειρο αριθμό βημάτων

Προσέξτε: δεν υπάρχει περιορισμος για το πότε συμβαίνουν αυτά τα γεγονότα – έτσι, δεν αποκλείονται αυθαίρετες καθυστερήσεις μηνυμάτων και αυθαίρετες διαφορές στις ταχύτητες των επεξεργαστών.

ΠΑΡΑΔΕΙΓΜΑ: ΥΠΕΡΧΕΙΛΙΣΗ ΜΗΝΥΜΑΤΩΝ

Θα περιγράψουμε ένα απλό αλγόριθμο υπερχείλισης μηνυμάτων σαν μια συλλογή από μηχανές πεπερασμένων καταστάσεων. Έστω ο επεξεργαστής p_i.

- *καταστάσεις:* η μεταβλητή χρώμα που είναι πράσινη ή κόκκινη, και οι προσωρινοί καταχωρητές μηνυμάτων (προς-τα-έσω και προς-τα-έξω).
- *αρχικές καταστάσεις:* για τον p_0, το χρώμα είναι πράσινο και όλοι οι προς-τα-έξω προσωρινοί καταχωρητές μηνυμάτων περιέχουν το μήνυμα M – για τους υπόλοιπους επεξεργαστές, το χρώμα είναι κόκκινο και όλοι οι προς-τα-έξω προσωρινοί καταχωρητές μηνυμάτων είναι κενοί.
- *μεταβάσεις:* αν το μήνυμα M βρίσκεται στον προς-τα-έσω προσωρινό καταχωρητή μηνυμάτων σου και το χρώμα σου είναι κόκκινο, τότε κάνε το χρώμα σου πράσινο και στείλε το μήνυμα M σε όλους τους προς-τα-έξω προσωρινούς καταχωρητές μηνυμάτων σου.

ΤΕΡΜΑΤΙΣΜΟΣ

Για τεχνικούς λόγους, οι νόμιμες εκτελέσεις ορίζονται ως άπειρες.

Συχνά όμως οι αλγόριθμοι τερματίζουν.

Για να ορίσουμε τον τερματισμό των αλγορίθμων, εισάγουμε τις *τερματικές καταστάσεις* των επεξεργαστών – καταστάσεις από τις οποίες οι επεξεργαστές δεν μπορούν να φύγουν.

Μια (νόμιμη) εκτέλεση έχει τερματίσει όταν όλοι οι επεξεργαστές έχουν μπει σε τερματικές καταστάσεις και δεν υπάρχουν καθόλου μετεπιβαζόμενα μηνύματα (σε προς-τα-έσω ή προς-τα-έξω προσωρινούς καταχωρητές μηνυμάτων).

ΜΕΤΡΑ ΠΟΛΥΠΛΟΚΟΤΗΤΑΣ

Αυτά αναφέρονται στη χειρότερη περίπτωση.

- *Πολυπλοκότητα Μηνυμάτων:* ο μέγιστος αριθμός μηνυμάτων που αποστέλλονται σε οποιαδήποτε νόμιμη εκτέλεση.
- *Χρονική Πολυπλοκότητα:* ο μέγιστος χρόνος μέχρι τον τερματισμό σε οποιαδήποτε νόμιμη εκτέλεση (πρώτη προσπάθεια).

Πώς όμως μετράμε το χρόνο;
Κατασκευάζουμε μια *χρονισμένη εκτέλεση* από μια εκτέλεση αποδίδοντας μη φθίνοντες χρόνους στα γεγονότα της εκτέλεσης έτσι ώστε:

- Ο χρόνος μεταξύ αποστολής και παραλαβής οποιουδήποτε μηνύματος είναι το πολύ 1.
- Κάθε επεξεργαστής έχει ένα υπολογιστικό βήμα αμέσως μετά από κάθε παραλαβή μηνύματος.

Αυτό κανονικοποιεί μια (νόμιμη) εκτέλεση έτσι ώστε η μέγιστη καθυστέρηση μηνύματος σε μια εκτέλεση να είναι 1 – εξακολουθεί να επιτρέπει αυθαίρετες αναμίξεις των γεγονότων.

- *Χρονική Πολυπλοκότητα:* ο μέγιστος χρόνος μέχρι τον τερματισμό σε οποιαδήποτε χρονισμένη νόμιμη εκτέλεση.

ΑΝΑΛΥΣΗ ΠΟΛΥΠΛΟΚΟΤΗΤΑΣ ΓΙΑ ΤΟΝ ΑΛΓΟΡΙΘΜΟ ΧΕΙΛΙΣΗΣ ΜΗΝΥΜΑΤΩΝ

Τερματική κατάσταση είναι εκείνη όπου το χρώμα κάθε επεξεργαστή είναι πράσινο.

- Η πολυπλοκότητα μηνυμάτων είναι $2m$, όπου m είναι ο αριθμός των ακμών του γράφου.
- Η χρονική πολυπλοκότητα είναι $diam(G) + 1$, όπου $diam(G)$ είναι η διάμετρος του γράφου G, δηλαδή το μήκος του μακρυτάτου μονοπατιού του.

ΣΥΓΧΡΟΝΟ ΜΟΝΤΕΛΟ ΑΝΤΑΛΛΑΓΗΣ ΜΗΝΥΜΑΤΩΝ

Οι επεξεργαστές εκτελούν συγχρονισμένους γύρους.

Ένας γύρος είναι μια ακολουθία γεγονότων της μορφής παραλαβή, ..., παραλαβή, βήμα, ..., βήμα.

- Διανέμονται όλα τα μηνύματα που βρίσκονται σε προς-τα-έξω καταχωρητές μηνυμάτων.
- Όλοι οι επεξεργαστές παίρνουν ένα υπολογιστικό βήμα ο καθένας.

Μια εκτέλεση είναι μια ακολουθία από γύρους. Μια εκτέλεση είναι νόμιμη αν είναι άπειρη. Κάθε νόμιμη εκτέλεση στο σύγχρονο μοντέλο θα καλείται σύγχρονη εκτέλεση.

Σε μια σύγχρονη εκτέλεση, ο χρόνος εκτέλεσης είναι ο αριθμός των γύρων μέχρι τον τερματισμό.

ΥΠΕΡΧΕΙΛΙΣΗ ΜΗΝΥΜΑΤΩΝ ΣΤΟ ΣΥΓΧΡΟΝΟ ΜΟΝΤΕΛΟ

Τι θα συμβεί αν ο αλγόριθμος υπερχείλισης μηνυμάτων εκτελεστεί στο σύγχρονο μοντέλο, πάνω σε ένα τρίγωνο;

- Πρώτος γύρος:
  - Το μήνυμα M παραλαμβάνεται στον p_1 από τον p_0.
  - Το μήνυμα M παραλαμβάνεται στον p_2 από τον p_0.
  - Ο p_0 δεν κάνει τίποτα.
  - Ο p_1 γίνεται πράσινος και στέλνει το μήνυμα M στους p_0 και p_2.
  - Ο p_2 γίνεται πράσινος και στέλνει το μήνυμα M στους p_0 και p_1.
- Δεύτερος γύρος:
  - Το μήνυμα M παραλαμβάνεται στον p_0 από τους p_1 και p_2.
  - Το μήνυμα M παραλαμβάνεται στον p_1 από τον p_2.
  - Το μήνυμα M παραλαμβάνεται στον p_2 από τον p_1.
  - Οι p_0, p_1 και p_2 δεν κάνουν τίποτα.

Η πολυπλοκότητα μηνυμάτων και η χρονική πολυπλοκότητα παραμένουν όπως και στο ασύγχρονο μοντέλο (δηλαδή, $2m$ και $diam(G)+1$, αντίστοιχα).

ΕΚΠΟΜΠΗ ΜΗΝΥΜΑΤΟΣ ΠΑΝΩ ΣΕ ΓΝΩΣΤΟ ΓΕΝΝΗΤΟΡΙΚΟ ΔΕΝΔΡΟ ΜΕ ΡΙΖΑ

Κάθε επεξεργαστής έχει τοπικές μεταβλητές γονέας και παιδί, οι οποίες δείχνουν ποια από τα προσπίπτοντα κανάλια οδηγούν στο γονέα και τα παιδιά του σε ένα γεννητορικό δένδρο του γράφου.

Επίσης, κάθε επεξεργαστής έχει μια λογική τοπική μεταβλητή που καλείται τερματισμένος.

- Η ρίζα στέλνει αρχικά το μήνυμα M στα παιδιά της.
- Όταν ένας επεξεργαστής λάβει το μήνυμα M από τον γονέα του, το στέλνει στα παιδιά του και τερματίζει.

Ποιες είναι οι πολυπλοκότητες στο σύγχρονο και το ασύγχρονο μοντέλο;

- Η χρονική πολυπλοκότητα είναι ίση με d, το βάθος του γεννητορικού δένδρου. (Σημείωση: $d \leq n-1$.)
- Η πολυπλοκότητα μηνυμάτων είναι ίση με $n-1$, αφού ένα ακριβώς μήνυμα αποστέλλεται πάνω σε κάθε ακμή του δένδρου.

ΣΥΛΛΟΓΗ ΜΗΝΥΜΑΤΩΝ ΠΑΝΩ ΣΕ ΓΝΩΣΤΟ ΓΕΝΝΗΤΟΡΙΚΟ ΔΕΝΔΡΟ ΜΕ ΡΙΖΑ

Το αντίθετο της εκπομπής μηνύματος από τη ρίζα: κάθε επεξεργαστής θέλει να στείλει ένα μήνυμα προς τη ρίζα.

- Κάθε φύλλο στέλνει το μήνυμά του προς το γονέα του.
- Κάθε μη-φύλλο περιμένει μέχρι να πάρει μήνυμα από κάθε ένα από τα παιδιά του. Τότε, τυλίγει μαζί όλα τα μηνύματα που έλαβε και τα στέλνει στο γονέα του (εκτός και αν είναι η ρίζα).

Η χρονική πολυπλοκότητα και η πολυπλοκότητα μηνυμάτων παραμένουν οι ίδιες.

ΚΑΤΑΣΚΕΥΗ ΓΕΝΝΗΤΟΡΙΚΟΥ ΔΕΝΔΡΟΥ ΜΕ ΠΡΟΚΑΘΟΡΙΣΜΕΝΗ ΡΙΖΑ

Είναι χρήσιμο να έχουμε ένα γεννητορικό δένδρο. Πως μπορούμε να το κατασκευάσουμε όταν έχουμε προκαθορίσει ως ρίζα ένα διακεκριμένο επεξεργαστή;

Τροποποιούμε τον αλγόριθμο υπερχείλισης μηνυμάτων ως εξής:

- Η ρίζα στέλνει το μήνυμα M σε όλους τους γείτονές της.
- Όταν ένας επεξεργαστής (άλλος από τη ρίζα) λάβει το μήνυμα M για πρώτη φορά, τότε κατονομάζει τον αποστολέα ως γονέα του, στέλνει το μήνυμα γονέας προς το γονέα του, και στέλνει το μήνυμα M σε όλους τους γείτονές του.
- Όταν ένας επεξεργαστής λάβει το μήνυμα M ξανά, τότε στέλνει το μήνυμα απόρριψη προς τον αποστολέα του μηνύματος.
- Κάθε επεξεργαστής χρησιμοποιεί τα μηνύματα γονέας και απόρριψη για να θέσει το γονέα του και τα παιδιά του, και να ξέρει πότε θα τερματίσει.

ΕΚΤΕΛΕΣΗ ΤΟΥ ΑΛΓΟΡΙΘΜΟΥ ΚΑΤΑΣΚΕΥΗΣ ΓΕΝΝΗΤΟΡΙΚΟΥ ΔΕΝΔΡΟΥ

- Στο σύγχρονο μοντέλο, ο αλγόριθμος παράγει πάντοτε ένα *δένδρο αναζήτησης κατά πλάτος*.
- Στο ασύγχρονο μοντέλο, ο αλγόριθμος δυνατό να μην παράγει ένα δένδρο αναζήτησης κατά πλάτος.

Και στις δύο περιπτώσεις, η πολυπλοκότητα μηνυμάτων είναι $O(m)$ και η χρονική πολυπλοκότητα είναι $O(diam(G))$.

ΑΛΓΟΡΙΘΜΟΣ ΚΑΤΑΣΚΕΥΗΣ ΓΕΝΝΗΤΟΡΙΚΟΥ ΔΕΝΔΡΟΥ ΑΝΑΖΗΤΗΣΗΣ ΚΑΤΑ ΒΑΘΟΣ ΜΕ ΓΝΩΣΤΗ ΡΙΖΑ

- όταν η ρίζα εκτελεί το πρώτο της υπολογιστικό βήμα ή ενας επεξεργαστής διάφορος από τη ρίζα λαμβάνει το μήνυμα M:
  - κατονόμασε τον αποστολέα ως γονέα (αν δεν είσαι η ρίζα)
  - για κάθε γειτονά σου
    - στείλε το μήνυμα M στο γειτονά σου
    - ανάμενε να λάβεις πίσω το μήνυμα γονέας ή το μήνυμα απόρριψη
  - στείλε το μήνυμα γονέας στο γονέα σου
- άν ένας επεξεργαστής λάβει το μήνυμα M σε οποιαδήποτε άλλη περίπτωση, τότε στέλνει το μήνυμα απόρριψη στον αποστολέα
- ο κάθε επεξεργαστής χρησιμοποιεί τα μηνύματα γονέας και απόρριψη για να προσδιορίσει τα παιδιά του και να τερματίσει.

Ο αλγόριθμος αυτός είναι αντίστοιχος με τον ακολουθιακό αλγόριθμο αναζήτησης κατά βάθος.
Η πολυπλοκότητα μηνυμάτων και η χρονική πολυπλοκότητα είναι και οι δύο $O(m)$.

ΑΛΓΟΡΙΘΜΟΣ ΚΑΤΑΣΚΕΥΗΣ ΓΕΝΝΗΤΟΡΙΚΟΥ ΔΕΝΔΡΟΥ ΖΩΡΙΣ ΓΝΩΣΤΗ ΡΙΖΑ

Υποθέτουμε ότι οι επεξεργαστές έχουν *μοναδικές ταυτότητες* (αλλοιώς, το πρόβλημα είναι αλγοριθμικά ανεπίλυτο!).

- *Ιδέα:* Κάθε επεξεργαστής χρησιμοποιεί τη μοναδική του ταυτότητα για να τρέξει ένα "αντίγραφο" του προηγούμενου αλγόριθμου κατασκευής γεννητορικού δένδρου αναζήτησης κατά βάθος με γνωστή ρίζα (τον εαυτό του).
- Κάθε φορά που δυο διαφορετικά "αντίγραφα" συγκρούονται, χρησιμοποιούνται οι μοναδικές ταυτότητες για να προσδιοριστεί ποιο θα νικήσει.

Η πολυπλοκότητα μηνυμάτων είναι $O(mn)$ και η χρονική πολυπλοκότητα είναι $O(m)$.