Σχεδίαση και Ανάλυση Αλγορίθμων

2η Εργασία : «Κατάταξη στοιχείων ταξινομημένων λιστών»

Java Threads

Documentation

Από τους φοιτητές:
Μποζιονέλος Στέφανος – ΜΠΣΠ13070
Παρασκάκης Ιωάννης – ΜΠΣΠ13086

Μποζιονέλος Στέφανος – ΜΠΣΠ13070
Παρασκάκης Ιωάννης – ΜΠΣΠ13086

1. Γενικά

Στην παρούσα εργασία κληθήκαμε να κατατάξουμε τα στοιχεία δύο(2) ταξινομημένων λιστών Χ και Υ σε σύστημα διαμοιραζόμενης μνήμης και με χρήση νημάτων για «παράλληλη» επεξεργασία. Για την υλοποίηση μας επιλέξαμε τη χρήση νημάτων της Java (java threads) και η ανάπτυξη έγινε στην πλατφόρμα του NetBeans.

Η υλοποίηση της εργασίας μας χωρίζεται σε δύο (2) φάσεις όπως υποδεικνύεται και από την εκφώνηση της εργασίας. Στη πρώτη φάση γίνεται η ν-αδική (όπου ν ο αριθμός των διαθέσιμων νημάτων) αναζήτηση επιλεγμένων στοιχείων της λίστας Χ στη λίστα Υ και (με χρήση αναδρομής) ο εντοπισμός της θέσης των εν λόγω στοιχείων στη λίστα Υ. Η δεύτερη φάση ξεκινάει με την ολοκλήρωση της $1^{ης}$ φάσης και αφού οι δύο (2) λίστες Χ και Υ έχουν υποδιαιρεθεί σε υπό-λίστες εκτελείται η συγχώνευση των υπό-λιστών. Κάθε μία εκ των συγχωνεύσεων αναλαμβάνεται από ένα νήμα στην παρούσα υλοποίηση.

2. Υλοποίηση

Για την παραπάνω υλοποίηση δημιουργήσαμε μία κύρια κλάση της java, μία κλάση που προσομοιώνει τα threads και τρεις (3) βοηθητικές κλάσεις. Πιο συγκεκριμένα:

- Η κύρια κλάση περιέχει τη μέθοδο main() που χρησιμοποιείται για να δημιουργηθεί το Master thread και να ξεκινήσει την εκτέλεση του. Τέλος στην κύρια κλάση ο χρήστης δίνει τον μέγιστο αριθμό thread που έχουμε στη διάθεση μας.
- Η κλάση που υλοποιεί τα thread (extends thread) είναι η κλάση που υλοποιεί το παράλληλο μας πρόγραμμα. Για τη δημιουργία ενός thread είναι απαραίτητο να οριστούν οι δύο λίστες μας (Χ και Υ) για να είναι γνωστές σε κάθε thread, ο τύπος του thread (Master/Slave/Backup) και το rank του κάθε thread. Το thread τύπου Master δημιουργεί τα threads τύπου slave. Τα threads τύπου Backup αναλύονται παρακάτω. Όλες οι διαδικασίες που είναι απαραίτητες για την εκτέλεση του προγράμματος γίνονται σε αυτή τη κλάση.
- Η πρώτη και σημαντικότερη βοηθητική κλάση είναι ο controller. Είναι η κλάση που διαχειρίζεται την κοινή μνήμη και αντιμετωπίζει τις πιθανές καταστάσεις αδιεξόδων ή ασυνέπειας που μπορεί να προκύψουν από τη ταυτόχρονη εκτέλεση του προγράμματος σε πολλά νήματα. Η κλάση αυτή περιέχει πολλές μεθόδους συγχρονισμού για κάθε βήμα του αλγορίθμου που είναι απαραίτητες για τη συνέπεια των δεδομένων. Ενδεικτικά:

- Set_number() & Get_number(): Η πρώτη καλείται από τον Master προκειμένου να δώσει στους slaves τον αριθμό που θα εκτελέσουν την ν-αδική αναζήτηση. Είναι φανερό ότι οποιαδήποτε πρόσβαση σε αυτή τη θέση μνήμης πριν την εκτέλεση από τον master θα προκαλούσε ασυνέπεια στο πρόγραμμα. Για το λόγο αυτό κλειδώνουμε τη δεύτερη με χρήση wait()/notifyAll() προκειμένου όλοι οι slaves να περιμένουν (wait()) μέχρι ο master να ολοκληρώσει την εκτέλεση του Set_number() και να τους αφυπνίσει (notifyAll()).
- Ομοίως αντιμετωπίζονται τα αδιέξοδα που προκύπτουν κατά την αναδρομή της αναζήτησης. Σε κάθε βήμα το εύρος του πίνακα που εκτελείται η αναζήτηση αλλάζει. Όταν οι slaves εκτελέσουν ένα βήμα σύγκρισης υποβάλουν μία αναφορά(report()) κατά την οποία δηλώνουν αν το στοιχείο που αναζητούν είναι μικρότερο ή μεγαλύτερο από το στοιχείο τους. Προκειμένου να προχωρήσουν στο επόμενο βήμα της αναζήτησης θα πρέπει να περιμένουν τον master να συλλέξει τα reports να τα επεξεργαστεί και να ορίσει το νέο εύρος αναζήτησης.

Οι μέθοδοι που υλοποιεί ο controller είναι:

- ❖ Register() : Καταγράφει ένα thread στο σύστημα.
- ❖ Set_number() : Ορίζει το στοιχείο αναζήτησης.
- ❖ Set_search_range() : Ορίζει το εύρος αναζήτησης.
- ❖ Grant_permissions_to_read() : Αφυπνήει τα thread που περιμένουν να λάβουν τα παραπάνω δεδομένα.
- ❖ Get_report() : Συλλέγει τα reports από τους slaves.
- ❖ Master_wait_reports() : Αναγκάζει τον Master να περιμένει να ολοκληρωθούν τα report.
- ❖ Get_number() : Οι slave παίρνουν το στοιχείο που έχει ορίσει ο master.
- ❖ Get_search_from() & Get_search_to() : Οι slave λαμβάνουν το εύρος αναζήτησης.
- ❖ Report() : Οι slave καταγράφουν τα αποτελέσματα τους.
- ❖ Barrier() : Οι slave περιμένουν τον master να ορίσει καινούριο εύρος αναζήτησης μετά τα reports.
- ❖ Άλλες μέθοδοι που επιστρέφουν τιμές Boolean για να γνωρίζουν τα thread πότε να τερματίσουν το κάθε βήμα.
- ❖ Μέθοδοι που συγκεντρώνουν και επιστρέφουν το αποτέλεσμα του merge.

- Οι άλλες δύο βοηθητικές κλάσεις είναι οι κλάσεις log και sublist. Η κλάση log διατηρεί τα reports ενώ η κλάση sublist διατηρεί τα δεδομένα για τις υπό λίστες που δημιουργούνται μετά τη $1^{η}$ φάση.

3. Οθόνες Εκτέλεσης

Εκτέλεση για 4 thread

```
How Many threads do you want to use
4
I am rank :2 Checking my number = 26 with the curent number = 15 from:0 to:20 in pointer :9
I am rank :4 Checking my number = 49 with the curent number = 15 from:0 to:20 in pointer :19
I am rank :1 Checking my number = 9 with the curent number = 15 from:0 to:20 in pointer :4
I am rank :3 Checking my number = 36 with the curent number = 15 from:0 to:20 in pointer :14
This Element has to be on the sublist = 1
I am rank :2 Checking my number = 17 with the curent number = 15 from:5 to:10 in pointer :6
I am rank :4 Checking my number = 26 with the curent number = 15 from:5 to:10 in pointer :9
I am rank :3 Checking my number = 19 with the curent number = 15 from:5 to:10 in pointer :7
I am rank :1 Checking my number = 16 with the curent number = 15 from:5 to:10 in pointer :5
Element 15 must enter in position 5
-----------------------------------------------------------------------------------------------
I am rank :2 Checking my number = 26 with the curent number = 45 from:0 to:20 in pointer :9
I am rank :4 Checking my number = 49 with the curent number = 45 from:0 to:20 in pointer :19
I am rank :1 Checking my number = 9 with the curent number = 45 from:0 to:20 in pointer :4
I am rank :3 Checking my number = 36 with the curent number = 45 from:0 to:20 in pointer :14
This Element has to be on the sublist = 3
I am rank :2 Checking my number = 42 with the curent number = 45 from:15 to:20 in pointer :16
I am rank :3 Checking my number = 44 with the curent number = 45 from:15 to:20 in pointer :17
I am rank :1 Checking my number = 41 with the curent number = 45 from:15 to:20 in pointer :15
I am rank :4 Checking my number = 49 with the curent number = 45 from:15 to:20 in pointer :19
This Element has to be on the sublist = 3
Element 45 must enter in position 17
-----------------------------------------------------------------------------------------------
```

Βλέπουμε τα βήματα της ν-αδικής αναζήτησης. Αρχικά τα thread ψάχνουν όλο τον πίνακα. Στέλνουν τα reports στον master και ο master αποφασίζει σε ποια υπό λίστα θα συνεχιστεί η αναζήτηση. Στη νέα υπό λίστα βλέπουμε ότι τα thread έχουν διαφορετικό βήμα αναζήτησης

```
I am rank :2 Checking my number = 26 with the curent number = 53 from:0 to:20 in pointer :9
I am rank :1 Checking my number = 9 with the curent number = 53 from:0 to:20 in pointer :4
I am rank :4 Checking my number = 49 with the curent number = 53 from:0 to:20 in pointer :19
I am rank :3 Checking my number = 36 with the curent number = 53 from:0 to:20 in pointer :14
Element 53 must enter at the end of the array
All next elements will be placed at the end of the array
All next elements will be placed at the end of the array
-----------------------------------------------------------------------------------------------
```

Αν για κάποιο στοιχείο εντοπιστεί ότι όλα τα στοιχεία είναι μικρότερα τότε όλη η υπό λίστα ανατίθεται μετά το τέλος του πίνακα ενώ για εξοικονόμηση χρόνου και πόρων δεν συνεχίζεται η αναζήτηση καθώς και κάθε επόμενη λίστα θα είναι στο τέλος του πίνακα αφού οι λίστες είναι ταξινομημένες.

Μποζιονέλος Στέφανος – ΜΠΣΠ13070
Παρασκάκης Ιωάννης – ΜΠΣΠ13086

```
array2.addAll(Arrays.asList(2,4, 7, 9, 9,16,17,19,21,26,28,29,31,34,36,41,42,44,46,49));
array1.addAll(Arrays.asList(5,7,10,11,15,18,19,22,35,45,46,46,50,52,53,60,62,64,66,68));
```

Έτσι για τους παραπάνω αρχικούς πίνακες έχουμε το παρακάτω αποτέλεσμα

```
Reports after Table division
From =5 To =17
From =19 To =19
From =0 To =5
From =19 To =19
Merged Table is
 | 2 | 4 | 5 | 7 | 7 | 9 | 9 | 10 | 11 | 15 | 2 | 4 | 7 | 9 | 9 | 16 | 17 | 18 | 19 | 19 | 21 | 22 | 26 | 28 | 29 | 35 | 45 | 46 | 46 | 50 | 52 | 53 | 60 | 62 | 64 | 66 | 68|
BUILD SUCCESSFUL (total time: 2 seconds)
```

Εκτέλεση για 5 thread:

```
run:
How Many threads do you want to use
5
I am rank :1 Checking my number = 9 with the curent number = 11 from:0 to:20 in pointer :3
I am rank :4 Checking my number = 41 with the curent number = 11 from:0 to:20 in pointer :15
I am rank :2 Checking my number = 19 with the curent number = 11 from:0 to:20 in pointer :7
I am rank :3 Checking my number = 29 with the curent number = 11 from:0 to:20 in pointer :11
I am rank :5 Checking my number = 49 with the curent number = 11 from:0 to:20 in pointer :19
This Element has to be on the sublist = 1
Element 11 must enter in position 3
-----------------------------------------------------------------------------------------------
I am rank :1 Checking my number = 9 with the curent number = 22 from:0 to:20 in pointer :3
I am rank :5 Checking my number = 49 with the curent number = 22 from:0 to:20 in pointer :19
I am rank :3 Checking my number = 29 with the curent number = 22 from:0 to:20 in pointer :11
I am rank :2 Checking my number = 19 with the curent number = 22 from:0 to:20 in pointer :7
I am rank :4 Checking my number = 41 with the curent number = 22 from:0 to:20 in pointer :15
This Element has to be on the sublist = 2
Element 22 must enter in position 7
-----------------------------------------------------------------------------------------------
I am rank :1 Checking my number = 9 with the curent number = 46 from:0 to:20 in pointer :3
I am rank :4 Checking my number = 41 with the curent number = 46 from:0 to:20 in pointer :15
I am rank :5 Checking my number = 49 with the curent number = 46 from:0 to:20 in pointer :19
I am rank :3 Checking my number = 29 with the curent number = 46 from:0 to:20 in pointer :11
I am rank :2 Checking my number = 19 with the curent number = 46 from:0 to:20 in pointer :7
This Element has to be on the sublist = 4
Element 46 must enter in position 15
-----------------------------------------------------------------------------------------------
```

```
I am rank :3 Checking my number = 29 with the curent number = 60 from:0 to:20 in pointer :11
I am rank :4 Checking my number = 41 with the curent number = 60 from:0 to:20 in pointer :15
I am rank :5 Checking my number = 49 with the curent number = 60 from:0 to:20 in pointer :19
I am rank :2 Checking my number = 19 with the curent number = 60 from:0 to:20 in pointer :7
I am rank :1 Checking my number = 9 with the curent number = 60 from:0 to:20 in pointer :3
Element 60 must enter at the end of the array
All next elements will be placed at the end of the array
All next elements will be placed at the end of the array
-----------------------------------------------------------------------------------------
Reports after Table division
From =7 To =15
From =19 To =19
From =3 To =7
From =0 To =3
From =19 To =19
Merged Table is
 | 2 | 4 | 5 | 7 | 7 | 10 | 11 | 2 | 4 | 7 | 9 | 15 | 18 | 19 | 22 | 2 | 4 | 7 | 9 | 9 | 16 | 17 | 19 | 35 | 45 | 46 | 46 | 50 | 52 | 53 | 60 | 62 | 64 | 66 | 68|
BUILD SUCCESSFUL (total time: 1 second)
```

4. Παρατηρήσεις

Στην εκτέλεση με τα 4 thread παρατηρούμε ότι το thread με id 2 παίρνει ένα πολύ μεγάλο κομμάτι του υπό πίνακα για να εκτελέσει τη συγχώνευση κάτι το οποίο δεν μας δίνει καλό speedup αφού το thread 2 αργεί να εκτελεστεί ενώ τα άλλα περιμένουν. Στην εκτέλεση με 5 thread παρόλο που τα 2 τελευταία (1 στην εκτέλεση με 4 thread) δεν λαμβάνουν υπό πίνακα ο διαμοιρασμός του πίνακα X είναι πιο συμμετρικός και παρατηρούμε καλύτερη κατανομή του πίνακα.

Δοκιμάσαμε την υλοποίηση μας σε εκτέλεση από 2 έως N/2 thread (N ο αριθμός των στοιχείων της λίστας X). Εξαιτίας της ψευδοπαραλληλίας της JVM όσον αφορά τα thread όσο αυξάναμε τα thread τόσο μεγαλύτερο χρόνο εκτέλεσης παρατηρούσαμε. Αυτό δε συμβαίνει σε πραγματικά κατανεμημένα συστήματα.

Όσον αφορά τα Back up thread, σε επέκταση της υλοποίησης μας, θα χρησιμοποιούσαμε τα back up threads για να αντιμετωπίσουμε τις «αδικίες» ανάμεσα στα threads. Έτσι όταν κάποιο thread ολοκλήρωνε το merge των λιστών του θα σκότωνε τον εαυτό του και θα επανεκκινούσε ως backup του thread με τον μεγαλύτερο φόρτο (μεγαλύτερες υπό λίστες) προκειμένου να ισομοιραστεί ο φόρτος και να μειωθεί ο χρόνος εκτέλεσης.