Προηγμένες Αρχιτεκτονικές Υπολογιστικών Συστημάτων 2015
Προγραμματιστική Εργασία #3

Θέμα

Στο εργαστήριο υλοποιήσατε τον αλγόριθμο Quicksort με τη βοήθεια των POSIX Threads (pthreads). Στο παράδειγμα εκείνο, τη διαδικασία δημιουργίας νέων threads αναλάμβανε το "πατρικό" thread, το οποίο κατασκεύαζε δυναμικά δύο νέα threads για να χειριστούν την αριστερή και δεξιά υπολίστα με τα μικρότερα και μεγαλύτερα του "οδηγού" (pivot) στοιχεία. Στο παράδειγμα του εργαστηρίου είδατε ότι η μέθοδος αυτή μπορεί να δημιουργήσει ανεξέλεγκτα μεγάλο αριθμό threads, γεγονός που επιβαρύνει σημαντικά την απόδοση της εφαρμογής.

Στην παρούσα άσκηση ζητείται να υλοποιήσετε τον αλγόριθμο Quicksort με μία δεξαμενή threads (thread pool). Στη δεξαμενή αυτή θα υπάρχει ένας σταθερός μικρός αριθμός threads (π.χ. 4), τα οποία θα δημιουργούνται στην αρχή του προγράμματος και θα τερματίζουν όταν θα έχει ολοκληρωθεί η ταξινόμηση. Τα threads θα αναλαμβάνουν πακέτα εργασίας από μια σφαιρική (global) ουρά εργασιών.

Ζητούμενο

α) Τροποποιήστε το παράδειγμα των pthreads + condition variables
(https://gist.github.com/mixstef/966be631a5d2601c4264#file-cv-example-c)
έτσι ώστε να υλοποιήσετε μια κυκλική ουρά με Ν θέσεις για μηνύματα (αντί για μία θέση, όπως στο παράδειγμα). Την ουρά αυτή θα χρησιμοποιήσετε ως σφαιρική ουρά εργασιών.

β) Αποφασίστε τα είδη των μηνυμάτων που θα στέλνετε στην ουρά και φτιάξτε την κατάλληλη struct για να τα φιλοξενήσει. Η ουρά θα χωράει Ν τέτοιες δομές.

Υπόδειξη:

- Κάθε πακέτο εργασίας θα διαμερίζει τον πίνακα με τον οποίο δουλεύει σε μικρότερα και μεγαλύτερα του pivot στοιχεία και θα στέλνει στην ουρά δύο **νέα μηνύματα με πακέτα εργασίας**. Στη συνέχεια, **δεν θα περιμένετε την ολοκλήρωση**: θα αναζητάτε νέα πακέτα εργασίας.
- Όπως και στο παράδειγμα του εργαστηρίου, όταν το μήκος του πίνακα είναι μικρότερο από ένα όριο, δεν κάνετε νέες αναθέσεις αλλά ολοκληρώνετε την ταξινόμηση επιτόπου. Μπορείτε να στείλετε ένα **μήνυμα ολοκλήρωσης** τμήματος του πίνακα (από-έως).
- Τα μηνύματα ολοκλήρωσης μπορεί να παρακολουθεί το main thread για να αποφασίσει πότε ολοκληρώθηκε η συνολική ταξινόμηση. Όταν αυτό συμβεί, ειδοποιήστε τα threads της δεξαμενής ότι πρέπει να τερματίσουν (**μήνυμα shutdown**).

γ) Κατασκευάστε τον κώδικα του κάθε thread: θα πρέπει να αναζητάτε νέα πακέτα εργασίας στην ουρά και να εκτελείτε το κομμάτι ταξινόμησης που περιέχουν -είτε στο ίδιο το thread είτε δημιουργώντας δύο νέα πακέτα εργασίας. Επίσης θα πρέπει να ελέγχετε για μηνύματα shutdown για να τερματίσετε το thread.

δ) Κατασκευάστε τον κώδικα του main(): αρχικοποιήστε την ουρά εργασιών, δημιουργήστε τα threads της δεξαμενής και τοποθετήστε στην ουρά το πρώτο πακέτο εργασίας. Στη συνέχεια, παρακολουθήστε την ουρά για μηνύματα ολοκλήρωσης (αθροίζοντας π.χ. τον αριθμό των ταξινομημένων στοιχείων). Όταν έχει ολοκληρωθεί η συνολική ταξινόμηση, στείλτε μήνυμα shutdown και περιμένετε στο join των threads. Τέλος, ελέγξτε την ορθότητα της ταξινόμησης, αποδεσμεύστε όλες τις δομές που

χρησιμοποιήσατε και τερματίστε την εφαρμογή.

Παραδοτέο

Θα πρέπει παραδώσετε (μέσω e-mail) α) το .c πρόγραμμά σας και β) αναφορά pdf με συνοπτική περιγραφή του κώδικά σας και τις πηγές που πιθανόν χρησιμοποιήσατε.

Προθεσμία παράδοσης: Τετάρτη 27/5/2015 10:00.