Παράλληλος Προγραμματισμός 2019
Προγραμματιστική Εργασία #2

(Προσοχή: η παράδοση της άσκησης θα γίνει μέσω github. Διαβάστε τις οδηγίες στο τέλος της εκφώνησης)

Θέμα

Στο εργαστήριο υλοποιήθηκε ο αλγόριθμος 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 δομή (C struct) για να τα φιλοξενήσει. Η ουρά θα χωράει Ν τέτοιες δομές.

Υποδείξεις:

- Κάθε thread όταν εκτελεί ένα πακέτο εργασίας θα **διαμερίζει** (partition) τον πίνακα με τον οποίο δουλεύει σε μικρότερα και μεγαλύτερα του pivot στοιχεία και θα στέλνει στην ουρά δύο **μηνύματα με τα νέα πακέτα εργασίας**. Στη συνέχεια, το thread **δεν θα περιμένει την ολοκλήρωση**, αντιθέτως θα αναζητά συνεχώς νέα πακέτα εργασίας.
- Όταν το μήκος του πίνακα είναι μικρότερο από ένα **όριο** (ορίστε το με #define), δεν κάνετε νέες αναθέσεις αλλά ολοκληρώνετε την ταξινόμηση επιτόπου. Όταν ένα thread ολοκληρώσει ένα πακέτο ταξινόμησης θα στέλνει ένα **μήνυμα ολοκλήρωσης** τμήματος του πίνακα (με στοιχεία: **από-έως**).
- Τα μηνύματα ολοκλήρωσης **θα 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 νέο πακέτο εργασίας στην ουρά και να εκτελείτε το κομμάτι ταξινόμησης που περιέχει – είτε στο ίδιο το thread, όταν το μέγεθος είναι κάτω από ένα όριο, 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.

Ως βάση για τον κώδικα του quicksort μπορείτε να χρησιμοποιήσετε το εξής παράδειγμα (σειριακής) υλοποίησης:
https://gist.github.com/mixstef/322145437c092783f70f243e47769ac6

<u>Παραδοτέο</u>
Η παράδοση θα γίνει μέσω github. Οδηγίες:

1. Αντιγράψτε (**fork**) το repository **https://github.com/mixstef/parprog1819a2** στο δικό σας repository. Βεβαιωθείτε οτι δουλεύετε αποκλειστικά στο **master branch**.
2. Τροποποιήστε κατάλληλα τα αρχεία που περιέχονται στο repository σας με το δικό σας περιεχόμενο:
    - Συμπληρώστε τα στοιχεία σας στο αρχείο **README.md** .
    - Βάλτε τον συνολικό κώδικά σας στο αρχείο **quicksort.c**.
    - Προσθέστε την αναφορά σας ως **report.pdf** .
    - **Προσοχή: πρέπει να διατηρήσετε τα ονόματα των παραπάνω αρχείων!**
3. Ενημερώστε το repository σας στο github εντός προθεσμίας. **Μην κάνετε pull request!**

Η εργασία είναι αυστηρά ατομική. Για την εγκυρότητα της υποβολής σας θα χρησιμοποιηθεί η χρονοσήμανση των αλλαγών (commits) των αρχείων σας.

Προθεσμία παράδοσης: Παρασκευή 31 Μαΐου, ώρα 23:59.