Σετ προγραμμάτων #1
(νήματα POSIX)

Πολλαπλασιασμός πινάκων με νήματα (50%)

Πρέπει να υλοποιήσετε και να χρονομετρήσετε παράλληλο πρόγραμμα για τον πολλαπλασιασμό τετραγωνικών πινάκων $N \times N$, χρησιμοποιώντας νήματα POSIX (PTHREADS) και *αυτοδρομολόγηση* (self-scheduling). Η κάθε εργασία θα πρέπει να είναι ο υπολογισμός ενός block (υποπίνακα) του αποτελέσματος, μεγέθους $S \times S$.

- Το πλήθος των εργασιών ισούται με τον συνολικό αριθμό των blocks (είναι, επομένως, συνάρτηση του S και του μεγέθους του πίνακα N).
- Το μέγεθος S του block (και άρα το πλήθος των εργασιών) θα πρέπει να είναι παραμετροποιήσιμο, έτσι ώστε να πειραματιστείτε με τον κόκκο παραλληλισμού (βλ. παρακάτω). Και, προφανώς, το S θα πρέπει να διαιρεί ακριβώς το μέγεθος του πίνακα.

Υλοποίηση απλού barrier (50%)

Καλείστε να υλοποιήσετε τον δικό σας μηχανισμό barrier και να τον συγκρίνετε με αυτό των νημάτων POSIX. Συγκεκριμένα, πρέπει να υλοποιήσετε τρεις συναρτήσεις που υλοποιούν το συγχρονισμό μεταξύ νημάτων μέσω κλήσεων φραγής:

- `barrier_init(b,n)`
- `barrier_wait(b)`
- `barrier_destroy(b)`

όπου `b` είναι δείκτης σε μία δομή `barrier_t` και `n` είναι το πλήθος των νημάτων που συμμετέχουν. Ο μηχανισμός σας θα πρέπει να στηρίζεται σε έναν μετρητή που μετράει πόσα νήματα περιμένουν και σε μεταβλητές συνθήκης (condition variables).

- Δεν επιτρέπονται καθολικές (global) μεταβλητές.
- Είναι σημαντικό να εξασφαλίσετε ότι η `barrier_wait(b)` δουλεύει ορθά *όταν κληθεί διαδοχικά δύο ή παραπάνω φορές*. Δηλαδή, πρέπει να συγχρονίζει σωστά όταν τα νήματα, αμέσως μετά την κλήση στον barrier, *κάνουν πάλι κλήση στον ίδιο barrier*.
- Δοκιμάστε στην υλοποίησή σας σε ένα δοκιμαστικό πρόγραμμα που παράγει νήματα, τα οποία επαναληπτικά (αρκετές φορές) κάνουν την παρακάτω διαδικασία: εκτελούν κάποιους υπολογισμούς και αμέσως μετά συγχρονίζονται με barrier.
- Τέλος, πειραματιστείτε με διαφορετικά πλήθη νημάτων, χρονομετρώντας το δοκιμαστικό πρόγραμμα τόσο με χρήση των δικών σας συναρτήσεων όσο και με τους barriers που παρέχουν τα νήματα POSIX.

Λεπτομέρειες

Βλ. στην πίσω σελίδα.

Απαιτούμενα

- Θα πρέπει να παραδώσετε πλήρη αναφορά, περιλαμβάνοντας και γραφικές παραστάσεις χρονομετρήσεων καθώς και συζήτηση γύρω από τα αποτελέσματα. Για τις κλήσεις φραγής απαιτείται *περιγραφή και επεξήγηση της λειτουργίας και της ορθότητας* καθώς και επισύναψη των προγραμμάτων δοκιμής.
- Τις απαντήσεις σας (πηγαίοι κώδικες + αναφορά) θα πρέπει να τα παραδώσετε με `turnin set1@mye023`. Πληροφορίες στην ιστοσελίδα του μαθήματος.
- Αν δεν γνωρίζετε, πρέπει να μάθετε να χρονομετράτε τον κώδικά σας στο linux (βλ. ιστοσελίδα μαθήματος για μια εισαγωγή). Για αυτό το σετ, η χρονομέτρηση να γίνει με χρήση της `gettimeofday()`.
- Τα προγράμματά σας για τον πολλαπλασιασμό πινάκων να τα δοκιμάσετε με 1, 2, 3, 4, 8 και 16 νήματα και να συγκρίνετε τους χρόνους με τον σειριακό κώδικα. Θα πρέπει να δοκιμάσετε 3 διαφορετικά πλήθη εργασιών (16, 256, 1024). Βλ. παρακάτω για δοκιμαστικούς πίνακες.
- Για κάθε περίπτωση, το πρόγραμμά σας θα εκτελείται τουλάχιστον 4 φορές και ο τελικός χρόνος θα είναι ο μέσος όρος των τεσσάρων χρόνων. Θα πρέπει επομένως να ετοιμάσετε έναν πίνακα αποτελεσμάτων για κάθε ένα από τα προγράμματά σας, π.χ. σαν τον παρακάτω:

Πολλαπλασιασμός πινάκων $N = 1024$ με 256 εργασίες

# νημάτων	1ο run	2ο run	3ο run	4ο run	Μέσος χρόνος
1					
2					
...					
16					

- Μαζί με τους παραπάνω πίνακες, θα πρέπει να ετοιμάσετε και γραφικές παραστάσεις των (μέσων) χρόνων συναρτήσει του αριθμού των νημάτων.

Παρατηρήσεις

1. Πληροφορίες σχετικά με το πως να χρονομετρήσετε στο Unix υπάρχουν στην ιστοσελίδα του μαθήματος, `http://www.cse.uoi.gr/~dimako/teaching/spring18.html`.
2. Η ανάπτυξη των προγραμμάτων σας μπορεί να γίνει οπουδήποτε αλλά η εκτέλεση και χρονομέτρηση των πειραμάτων σας θα πρέπει να γίνει σε υπολογιστές του τμήματος οι οποίοι διαθέτουν 4-πύρηνους επεξεργαστές (π.χ. `opti7020ws08`).
3. Τα αντίστοιχα σειριακά προγράμματα μπορείτε να τα βρείτε στην ιστοσελίδα του μαθήματος.
4. Για τον πολλαπλασιασμό πινάκων, δοκιμάστε πίνακες *ακεραίων* 1024×1024 στοιχείων (δίνονται στην ιστοσελίδα του μαθήματος). Τα στοιχεία τους θα πρέπει να τα αποθηκεύσετε σε αρχεία ώστε να μπορέσετε να ελέγξετε τη σωστή λειτουργία των προγραμμάτων σας συγκρίνοντας με τα αποτελέσματα του σειριακού (εγγυημένα σωστού) προγράμματος. **Προσοχή**: *μην χρονομετρήσετε το μέρος του κώδικα που διαβάζει / γράφει στα αρχεία*.

Προθεσμία παράδοσης:

Δευτέρα, 2 Απριλίου 2018

Βασίλειος Δημακόπουλος