Σετ Εργασιών 1
Ομάδα 6

Γιαννούκκος Παναγιώτης 2280
Χωροπανίτης Πασχάλης 2453

Άσκηση 1.1 - Σκέψη

- Για την υλοποίηση αυτής της εργασίας χρησιμοποιήσαμε ένα πίνακα από χαρακτήρες ως αγωγό σε μορφή circular buffer και δύο ακέραιους που υποδεικνύουν τις θέσεις εγγραφής και ανάγνωσης από και προς τον buffer. Αρχικά όλες οι θέσεις του αγωγού αρχικοποιούνται με το χαραχτηρα ‘\0’.
- Το νήμα εγγραφής παίρνει είσοδο από ένα αρχείο, ένα byte κάθε φορά, και γράφει στον αγωγό μόνο σε θέση που έχει σαν περιεχόμενο το ‘\0’ και μετακινήτε μια θέση μπροστά. Όταν βρίσκετε σε θέση με οποιοδήποτε άλλο χαρακτήρα, περιμένει εκέι μέχρι να “αδείασει” αυτή η θέση. Αν φτάσει σε θέση μεγαλύτερη του μεγέθους του αγωγού, τότε επιστρέφει στην αρχή.
- Όταν δεν υπάρχουν άλλα δεδομένα στο αρχείο εισόδου, καλείτε η pipe_writeDone() και θέτουμε το write_pos σε -1 για τη σηματοδότηση τέλους της εισόδου.

Άσκηση 1.1 – Σκέψη (Συνέχεια)

- Το νήμα ανάγνωσης διαβάζει μόνο όταν η θέση που βρίσκετε είναι άνιση του ‘\0’ και δεν έχει κληθέι η pipe_writeDone() (write_pos != -1). Αφού διαβάσει έναν χαρακτήρα από τον αγωγό, τον γράφει στο αρχείο εξόδου μέσω ενός file descriptor και θέτει το pipe_array[read_pos] = ‘\0’. Όταν δεν υπάρχουν άλλα δεδομένα για διάβασμα, το read_pos γίνεται -1 για να σηματοδοτήσει το main thread ότι τέλειωσε.
- Το main thread περιμένει να γίνει το read_pos = -1 και έπειτα να τερματίσει.

Άσκηση 1.1 - Υλοποίηση

```
thread_write {
    while(!EOF) {
        while(*write_pos != ‘\0’)  { nop; }
        pipe_write(char);
    }
    pipe_writeDone();
}
```

```
thread_read {
    while(1) {
        while(*read_pos==’\0’ &&
                write_pos!=-1) { nop; }
        res=pipe_read(&char);
        if(res=1) {write to output}
        else { break; }
    }
}
```

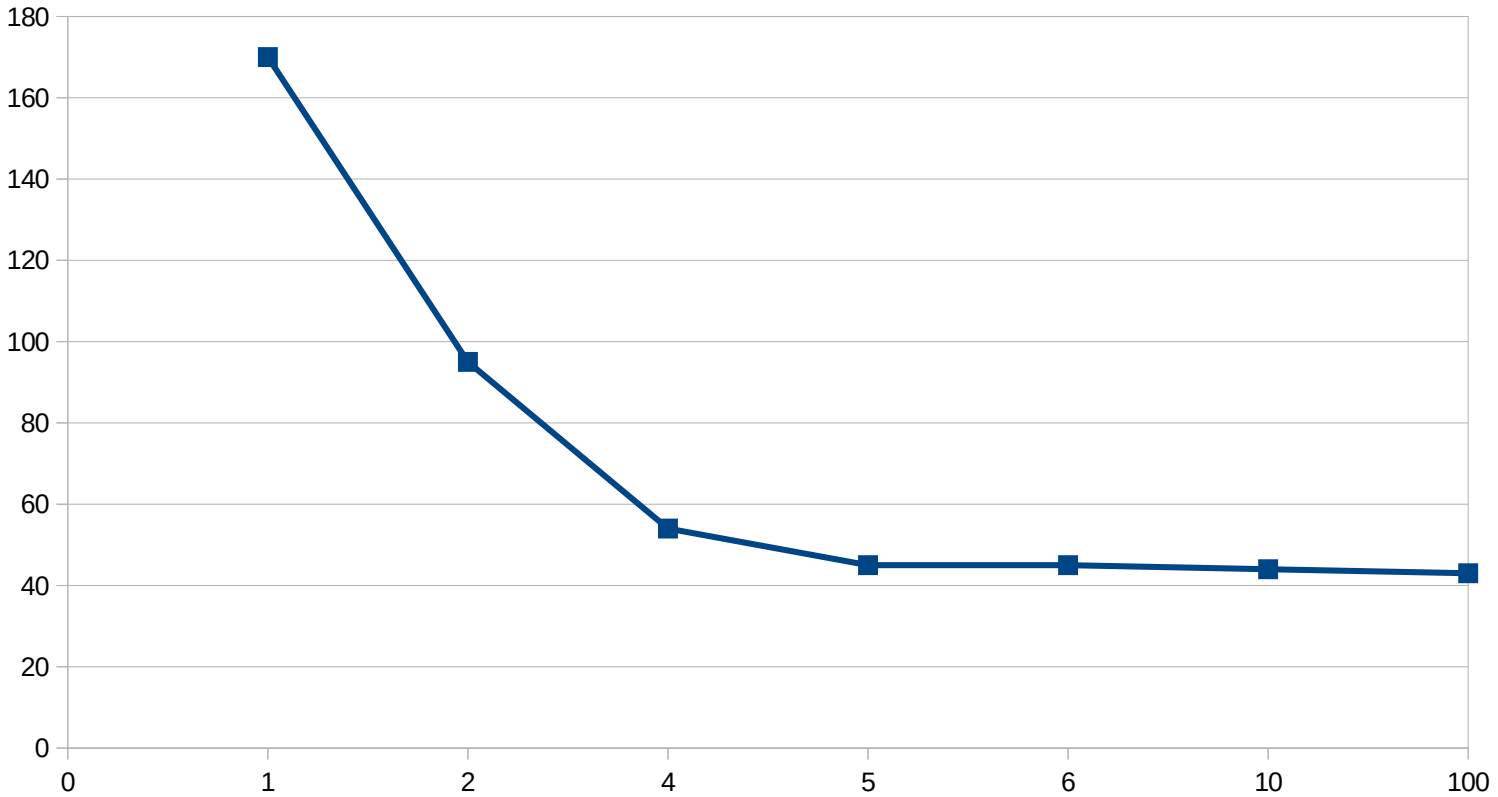
Άσκηση 1.2 - Σκέψη

- Αρχικά, δημιουργίσαμε ένα struct worker_t όπου αποθηκεύει πληροφορία για την κατάσταση του κάθε worker thread.
- Το main thread φτιάχνει ένα πίνακα με τέτοια structs όσα είναι και τα thread workers και κατα τη δημιουργία του κάθε worker, του δίνουμε σαν παράμετρο το index του σε αυτόν τον πίνακα.

Άσκηση 1.2 - Παρατηρήσεις

- Τις μετρήσεις τις πήραμε με τη χρήση της εντολής time του λειτουργικού Linux.
- Χρησιμοποιήσαμε για είσοδο 100 πρώτους αριθμούς απο 472,882,049 μέχρι 472,884,059.
- Παρατηρήσαμε ότι υπάρχει μεγάλη διαφορά στο χρόνο εκτέλεσης όταν δημιουργούμε λιγότερους workers από τα threads του επεξεργαστή. Όταν οι workers είναι περισσότεροι από τα threads του επεξεργαστή, τότε δεν υπάρχει μεγάλη διαφορά στο χρόνο τρεξίματος.

Άσκηση 1.2 - Μετρήσεις



Άσκηση 1.3 - Σκέψη

- Αρχικά φτιάξαμε ένα πίνακα από τυχαίους αριθμούς προς ταξινόμηση.
- Φτιάξαμε ένα struct όπου κρατάει πληροφορία για το μέρος του πίνακα που πρέπει να ταξινομήσει το κάθε thread.
- Το main thread ξεκινά την ταξινόμηση και κάθε φορά που ένα στοιχείο του πίνακα μπαίνει στη θέση του, τότε δημιουργουμε άλλα δύο threads για το αριστερό και δεξιό κομμάτι του πίνακα και η συνάρτηση εκτελείται αναδρομικά.

Άσκηση 1.3 - Υλοποίηση

```
quick_sort() {
    If (array size < 2) return;
    sort_section();
    Create two threads for left and right sections
    wait until both threads have finished
}
```