ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
Τμήμα Πληροφορικής και Τηλεπικοινωνιών
2η Εργασία - Τμήμα: Άρτιων Αριθμών Μητρώου
K24: Systems Programming – Εαρινό Εξάμηνο ‘20

Κωστόπουλος Σταύρος
Α.Μ.: 1115201700068

Η υλοποίηση του προγράμματος της 2ης Εργασίας, προκύπτει από την σύνθεση εφτά (7) πηγαίων αρχείων της μορφής .c (Το πρόγραμμα αναπτύχθηκε στην γλώσσα προγραμματισμού C).Τα εφτά αυτά αρχεία είναι τα παρακάτω:

- main.c
- functions.c
- hashtable.c
- kouvades.c
- queries.c
- tree.c
- topk.c
- worker.c
- dateandlist.c

Τα πηγαία αρχεία και τα αρχεία βιβλιοθήκης είναι γεμάτα με επεξηγηματικά σχόλια που εξηγούν τα βήματα και τις συναρτήσεις.

Παρακάτω ακολουθούν μερικές σημειώσεις που επεξηγούν την υλοποίηση της εργασίας:

1.Στην εκκίνηση του προγράμματος δημιουργούνται 2(δύο) NAMED FIFO PIPES ανάμεσα σε κάθε worker και στον diseaseAggregator.Τα pipes με όνομα pipenumberx (οπού x ο αριθμός του κάθε worker) εξυπηρετούν την επικοινωνία του diseaseAggregator προς τους Workers,ενώ τα pipes με όνομα worker2aggx (οπού x ο αριθμός του κάθε worker) εξυπηρετούν την επικοινωνία του worker προς τον diseaseAggregator.

2.Οι παραμέτροι (arguments) που χρησιμοποιούνται για την κλήση του εκτελέσιμου προγράμματος worker,είναι οι εξής:

./worker workernumber buffersize input_dir

3.Ο διαμοιρασμός των φακέλων επιτυγχάνεται με round-robin μέθοδο.

4.Οι δομές δεδομένων που χρησιμοποιήθηκαν στον worker είναι οι ίδιες με αυτές της πρώτης εργασίας.

5.Ο worker επομένως,δέχεται το όνομα ενός φακέλου,διαβάζει ένα ένα τα αρχεία του -αφού πρώτα τα ταξινομήσει με χρονολογική σειρά- και παίρνοντας μία μία τις εγγραφές των αρχείων,τις ελέγχει και αν δεν προκύψει κάποιο σφάλμα από τα στοιχεία της εγγραφής,την αποθηκεύει στις δομές δεδομένων του.Ύστερα από την ανάγνωση ενός αρχείου,ο worker στέλνει τα Summary statistics του αρχείου μέσω του worker2aggx pipe στον diseaseAggregator, ο οποίος με την χρήση της συνάρτησης select(),οργανώνει χρονικά τα μηνύματα και τα εκτυπώνει.

6.Αφού ολοκληρωθεί η συγκέντρωση των summary statistics όλων των αρχείων,το πρόγραμμα τίθεται σε αναμονή για να λάβει οποιαδήποτε από τις εντολές.Μόλις λάβει μία εντολή,ο diseaseAggregator την στέλνει μέσω των pipes στους workers και αυτοί με την σειρά τους επιστρέφουν κάποιο αποτέλεσμα.Ο diseaseAggregator,συγκεντρώνει τα αποτελέσματα από όλους τους workers,και εκτυπώνει.

7.Στην εντολή /exit ,ο diseaseAggregator στέλνει ένα SIGQUIT signal σε όλους τους worker,ώστε αυτοί να τερματίσουν αφού πρώτα δημιουργήσουν το log_file.xxx τους και απελευθερώσουν όλη την δεσμευμένη μνήμη.

8.Στο log_file κάθε worker καταγράφονται οι χώρες που ανέλαβε,καθώς και οι αριθμοί των συνολικών εγγραφών που έγινε προσπάθεια να αποθηκευτούν,των εγγραφών που αποθηκεύτηκαν επιτυχώς και των εγγραφών που απέτυχαν να αποθηκευτούν.

9.Για την ασφαλή read και write στα pipes,διαμορφώθηκαν δύο συναρτήσεις read_mesaage() και write_message() (βλ. functions.c).Πριν από οποιαδήποτε write,γράφεται πρώτα στο pipe ο αριθμός των bytes που πρόκειται να γραφτούν.Έτσι πριν από οποιαδήποτε read,μπορούμε να ξέρουμε τον αριθμό των bytes των δεδομένων που πρόκειται να διαβάσουμε,και αν αυτό είναι μεγαλύτερο του buffersize,το μήνυμα διαβάζεται σε μικρότερα buffesize byte sized κομμάτια,και αυτά ενώνονται ώστε να διαμορφώσουν το αρχικό μας μήνυμα.

ΣΗΜΑΝΤΙΚΟ: Το πρόγραμμα είναι αρκετά ασταθές.Ο diseaseAggregator συλλέγει τα summary stats επιτυχώς ώστε να προχωρήσει παρακάτω το πρόγραμμα(σε εντολές και signals) με την αλλαγή μιας printf για παράδειγμα όμως, παύει να τα δέχεται με επιτυχία με αποτέλεσμα το πρόγραμμα να blockάρει.Κατέβαλα πάρα πολύ χρόνο και προσπάθεια για την αντιμετώπιση αυτού του προβλήματος,με καμία παρόλα αυτά ουσιώδη επιτυχία.Με τράβηξε απίστευτα πίσω χρονικά,για αυτό ζητώ την κατανόησή σας
για τυχόν ατημέλητα κομμάτια κώδικα.Δεν είναι σε καμία περίπτωση αυτό που φανταζόμουν και ήθελα να παραδώσω.

Λόγω αυτού δεν κατάφερα να προχωρήσω παρακάτω για τα signals, αλλά επιχείρησα να υλοποιήσω το SIGQUIT signal στην εντολή /exit (βλ σημείωση 7 πιο πάνω) και σε ορισμένες “τυχερές” εκτελέσεις να μπορέσω να διαπιστώσω ότι δουλεύει ορθά.Ο κατάλογος που παραδίδω περιέχει μία απλή έκδοση του προγράμματος όπου τα πρωτόκολλα επικοινωνίας συλλέγουν επιτυχώς τα summary stats.Εμπεριέχεται ωστόσο και ακόμη ένας κατάλογος με

τον ακριβώς ίδιο κώδικα,αλλά με υλοποιημένο το SIGQUIT signal στην εντολή exit.Το πρόγραμμα χωρίς το SIGQUIT signal τρέχει επιτυχώς και οι workers απελευθερώνουν την δεσμευμένη μνήμη και τερματίζουν με την κλήση της /exit , και μετά από αυτούς το ίδιο κάνει και ο diseaseAggregator.

Υποσημείωση: Είχα να κλάψω από τον αποκλεισμό της Μπαρτσελόνα από την Λίβερπουλ.