ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΉ ΆΣΚΗΣΗ 2

Νικολακόπουλος Νικόλαος

sdi1800133

Για την υλοποίηση με τα pipes χρησιμοποίησα unnamed pipes. Για κάθε ζεύγος πατέρα- παιδιού χρησιμοποιήσα 1 unnamed pipe , αφού κάθε πατέρας πρέπει να διαβάζει από το ένα άκρο και το παιδί πρέπει να γράφει από το άλλο άκρο του pipe. Δεν χρειαζόταν στην υλοποίησή μου να έχουμε 2 unnamed pipes για κάθε ζεύγος αφού δεν είναι αναγκαίο ο πατέρας να γράφει και το παιδί να διαβάζει, αφού τις απαραίτητες πληροφορίες τις περνώ σαν argument στην exec. Οπότε για κάθε πατέρα έχω n unnamed pipes, όπου n = branch factor = ο αριθμός των παιδιών.

Χρησιμοποιώ την συνάρτηση poll ώστε να περιμένουν οι γονείς-διεργασίες μέχρι να τους γράψουν τα παιδιά κάποια πληροφορία μέσω pipes και τις αντιμετωπίζουν αντίστοιχα. Έτσι, το event που περιμένουν είναι προφανώς POLLIN.

Από το leaf κάνω write μέσω του pipe τον κάθε ένα πρώτο και τον χρόνο που χρειάστηκε για να βρεθεί και το στέλνω σαν string στον internal σε μορφή «prime,time». Έπειτα στον internal εισάγω ταξινομημένα το Prime και το time σε μια λίστα. Όταν ο leaf τελειώσει στέλνει ένα string της μορφής «done,time» , όπου time είναι ο χρόνος που χρειάστηκε ο συγκεκριμένος worker για να ολοκληρώσει την δουλειά του. Στο internal process αποθηκεύω σε έναν πίνακα τους χρόνους των Wi , δηλαδή των leaf processes που δημιούργησε οπότε μπορούμε εύκολα να αποκτήσουμε πρόσβαση σε αυτούς με αύξουσα σειρά του δείκτη i. Στη συνέχεια, παίρνουμε τον κάθε πρώτο αριθμό και τον χρόνο του από την ταξινομημένη λίστα και τα στοιχίζουμε με σειρά σε έναν buffer (σε μορφή prime1,time1 prime2,time2 …), μέχρι να γεμίσει το buffer. Όταν γεμίσει το στέλνουμε στην ρίζα και επαναλαμβάνουμε την ίδια διαδικασία μέχρι να αδειάσει η λίστα. Τότε, στέλνουμε μήνυμα της μορφής done,timeW1,timeW2,… , όπου timeWi είναι ο χρόνος του κάθε leaf process και είναι σορταρισμένοι με αύξουσα σειρά i , όπου i είναι index στα workers=leaf(όταν λέω worker εννοώ leaf). Η ρίζα αφού πάρει το batch από primes,times μέσω του pipe τα αποθηκεύει σε έναν πίνακα από strings , τοποθετώντας τα στην θέση i, όπου i είναι το i-οστό παιδί της ρίζας και γι’αυτό μετά μπορούμε να τους κάνουμε Print εύκολα για κάθε array[i]. Όταν, λάβουμε μήνυμα της μορφής done,time1,time2,… μέσω της strtok τοποθετούμε σε έναν δισδιάστατο πίνακα τους χρόνους (της μορφής Wtimes[i][j] , όπου i , η internal process και j , η leaf process), οπότε μετά είναι επίσης εύκολο να κάνουμε Print όλους τους χρόνους των workers. Το να βρω τους min Worker time και τον αντίστοιχο max είναι αρκετά τετριμμένο.

Για να φτιάχνω τα παιδιά έχω μια συνάρτηση spawn childrens στο αρχείο functions.c , στην οποία περνάω σαν όρισμα:

το lower bound και το Upper bound του πατέρα, το διάστημα που πρέπει να μοιράσει σε κάθε παιδί, το οποίο ισούται με το upper bound-lower bound του πατέρα διά τον αριθμό των παιδιών ( της μορφής (ub-lb)/n). Ωστόσο κάποιες φορές μπορεί να έχει υπόλοιπο η διαίρεση αυτή, οπότε θα το προσθέσουμε στο διάστημα που πρέπει να διαχειριστεί ο τελευταίος worker. Ακόμα, περνάμε σαν όρισμα το branch factor και τον πίνακα με τα fd των pipes , ώστε να το περάσουμε σαν argument στην exec για να γράφει το κάθε παιδί στο κατάλληλο pipe. Επίσης περνάμε ένα flag για το αν θέλουμε να κάνουμε spawn leaf ή internal nodes, ώστε να εκτελέσουμε το ανάλογο exec και να περάσουμε το κατάλληλο αριθμό για να εκτελεστεί στο leaf node το ανάλογο prime algorithm. Τέλος, περνάω σαν όρισμα το process id της ρίζας για να της στέλνουμε signals από τα leaf nodes στο root.

Στο εκτελέσιμο Primes βρίσκουμε τα ανάλογα flags και καλούμε τη συνάρτηση root η οποία εκτελεί τον κώδικα της ρίζας.