

Κωνσταντίνος Κανελλάκης
Αριθμός Μητρώου: 1115202000064
Εργασία 1-Λειτουργικά Συστήματα

Για την ανάλυση της λειτουργίας του κώδικα της πρώτης εργασίας του μαθήματος θα γίνει επεξήγηση ανά αρχείο με εξαίρεση των αρχείων `main.c` και `execchild.c` για τα οποία κρίνεται αναγκαίος ο σχολιασμός τους παράλληλα σε ορισμένα σημεία. Πρέπει επίσης να σημειωθεί ότι η μέτρηση των γραμμών ξεκινάει από 0. Συνεπώς σε περίπτωση που ζητηθεί η γραμμή 1 του αρχείου αυτό αντιστοιχεί στην γραμμή 0 στο πρόγραμμα. Αντίστοιχα και με τα τμήματα που χωρίζεται το κείμενο.

functions.h

- Struct semaphore

Στην δομή αυτή αποθηκεύονται δύο σεμαφόροι και ένας αριθμός τύπου `int`. Ο `num` είναι ένας μετρητής αιτημάτων και αρχικοποιείται με 0. Ο `sp` ένας τυπικός σεμαφόρος που μπλοκάρει τα αιτήματα, και αρχικοποιείται με 0. Ο `mutex` θα χρησιμοποιηθεί ως `binary semaphore` (του οποίου η λειτουργία ταυτίζεται με αυτή του `mutex`) εξού και το όνομα του, ενώ αρχικοποιείται με 1. Η μεταβλητή `mutex` ορίζεται κυρίως για να προστατεύει τις αλλαγές στον μετρητή `num`.

- Struct memory

Στην δομή αυτή περιέχονται μεταβλητές οι οποίες κρατάνε πληροφορίες για το κείμενο και τα τμήματα του καθώς και μερικούς σεμαφόρους για την επικοινωνία των παιδιών με τον πατέρα. Στην μεταβλητή `segm` αποθηκεύεται το τμήμα το οποίο ζητείται από το παιδί, στην μεταβλητή `total` τον αριθμό των αιτημάτων που ολοκληρώθηκαν, στην μεταβλητή `requests` τον αριθμό των αιτημάτων που πρέπει να κάνει κάθε παιδί, στην μεταβλητή `last_line` είτε 0 αν οι γραμμές του αρχείου διαιρούνται χωρίς υπόλοιπο με τον αριθμό των γραμμών ανά τμήμα που δίνεται ως όρισμα είτε με το υπόλοιπο αυτής της διαίρεσης που υποδηλώνει το πλήθος των γραμμών στο τελευταίο τμήμα. Τέλος αποθηκεύουμε στην μεταβλητή `line_seg` τον αριθμό των γραμμών ανά τμήμα και στην `total_segs` τον αριθμό των τμημάτων στον οποίο χωρίζεται το κείμενο.

- Τέλος ορίζονται και δύο μεταβλητές `smphr` και `mem` που αντιστοιχούν σε δείκτες στις παραπάνω δύο δομές.

Timer.h

Το αρχείο αυτό προέρχεται από το μάθημα Παράλληλα Συστήματα όπου έχει δωθεί ως βοηθητικός κώδικας για την 1^η εργασία. Χρησιμοποιείται για την χρονομέτρηση των αιτημάτων, όπου αυτό έχει ζητηθεί στην εργασία.

functions.c

5 συναρτήσεις υλοποιούνται στο αρχείο οι οποίες χρησιμοποιούνται τόσο από την μητρική διεργασία όσο και από τα παιδιά διεργασίες. Η `get_key()` επιστρέφει το κλειδί του `shared memory` που χρησιμοποιούν όλες οι διεργασίες για να επικοινωνούν. Η `read_file()` επιστρέφει μια γραμμή από το

Κωνσταντίνος Κανελλάκης
Αριθμός Μητρώου: 1115202000064
Εργασία 1-Λειτουργικά Συστήματα

αρχείο που δείχνει ο δείκτης `fp`. Η `init()` αρχικοποιεί τα στοιχεία της ενός αντικειμένου τύπου `struct memory` ενώ οι επόμενες δύο συναρτήσεις κάνουν `post` και `wait` αντίστοιχα τον σεμαφόρο του αντικειμένου.

main.c

Το αρχείο αυτό αποτελεί την υλοποίηση της μητρικής διεργασίας. Είναι άξιο να σημειωθεί ότι τα περιεχόμενα του αρχείου κειμένου που δέχεται ως όρισμα αποθηκεύονται στην μεταβλητή `array` όπου αποτελεί έναν δυσδιάστατο πίνακα `char`. Κάθε γραμμή `i` του πίνακα αποθηκεύεται στην αντίστοιχη `array[i]` θέση του πίνακα. Με την βοήθεια αυτού του πίνακα αντιγράφεται κάθε φορά το τμήμα που ζητείται στην κοινόχρηστη μνήμη.

execchild.c

Για να εξασφαλίσουμε την τυχαιότητα των τιμών ανά διεργασία παιδί αρχικοποιούμε την μεταβλητή `seed` ως την τιμή της ώρας στην οποία τρέχει ο κώδικας συν το `id` της διεργασίας. Το `seed` στέλνεται αργότερα ως όρισμα στην συνάρτηση `rand_r()`. Η κοινόχρηστη μνήμη συνδέεται στη διεργασία παιδί και αν δεν υπάρξει κάποιο σφάλμα η εκτέλεση συνεχίζεται.

main.c && execchild.c

Αφού η μητρική διεργασία «συνδεθεί» με την κοινόχρηστη μνήμη και κάνει τις απαραίτητες αρχικοποιήσεις, δημιουργούνται `N` διεργασίες με την συνάρτηση `fork()`, οι οποίες καλούνται να εκτελέσουν τον κώδικα στο αρχείο `execchild.c` με όρισμα το `id` του καθενός.

Ύστερα κάθε διεργασία παιδί αφού κάνει τις απαραίτητες τακτοποιήσεις και προετοιμασίες ξεκινά να παράει αιτήματα εντός της `for`. Εφόσον η διεργασία παιδί επιλέξει τυχαία με την βοήθεια της συνάρτησης `rand_r()` την γραμμή του τμήματος που θέλει να πάρει προχωρά στην αύξηση του μετρητή `sr[x].num` όπου `x` είναι ο αριθμός του τμήματος στο οποίο βρίσκεται η γραμμή.

Εδώ είναι σκόπιμο να αναφερθεί ότι για την διαχείριση των αιτημάτων δημιουργείται και αποθηκεύεται στην κοινόχρηστη μνήμη ένας πίνακας από αντικείμενα τύπου `struct semaphore` στον οποίο αναφερόμαστε μέσω του δείκτη `sp`. Ο αριθμός των θέσεων του πίνακα ταυτίζεται με τον αριθμό των τμημάτων στα οποία έχει χωριστεί το κείμενο που δίνεται ως όρισμα στην μητρική διεργασία. Συνεπώς όταν αναφερόμαστε στο στοιχείο του πίνακα `sr[x]` κρατάμε πληροφορίες για τις αιτήσεις που θέλουν να πάρουν γραμμή από το τμήμα `x`.

Όπως γίνεται αντιληπτό ο μετρητής `sr[x].num` κρατάει τον αριθμό των διεργασιών που περιμένουν να πάρουν γραμμή από το τμήμα `x`. Όταν αυξάνουμε ή μειώνουμε τον μετρητή, αυτός προστατεύεται από τον `sr[x].mutex` ώστε να αποφευχθεί το `race condition` στην μεταβλητή αυτή.

Κωνσταντίνος Κανελλάκης
Αριθμός Μητρώου: 1115202000064
Εργασία 1-Λειτουργικά Συστήματα

Αφού η διεργασία παιδί ενημερώσει το `sp[x].num` τότε υπάρχουν δύο περιπτώσεις για το τι θα ακολουθήσει για αυτή την διεργασία.

1. Αν το `sp[x].num` ισούται με 1 αυτό σημαίνει ότι είναι η πρώτη διεργασία που αιτείται το `x` τμήμα και άρα περιμένει στον σεμαφόρο `k->next`.
2. Αν το `sp[x].num` δεν ισούται με 1 αυτό σημαίνει ότι ήδη μία διεργασία περιμένει για το τμήμα `x` και άρα θα εξυπηρετηθούν μαζί χωρίς αμοιβαίο αποκλεισμό όταν περάσει η πρώτη διεργασία. Συνεπώς μπλοκάρεται από τον σεμαφόρο `sp[x].sp`.

Με αυτόν τον τρόπο κάθε διεργασία που ζητά πρώτη ένα τμήμα περιμένει την σειρά της στον σεμαφόρο `k->next` ενώ αυτές που ζητάνε το ίδιο τμήμα και έρχονται αργότερα περιμένουν στον `sp[x].sp` μέχρι να τους δωθεί το σήμα από την πρώτη διεργασία(την πρώτη για κάθε αιτούμενο `x`) που θα είναι υπεύθυνη να επικοινωνήσει με την μητρική διεργασία για να πάρουν την γραμμή που θέλουν.

Από την πλευρά της, η μητρική διεργασία μπαίνοντας στην `while` καθαρίζει την μεταβλητή `str` που θα αποθηκευτεί το τμήμα και ανεβάζει τον σεμαφόρο για το `k->next` έτσι ώστε να περάσει η πρώτη διεργασία παιδί και να αιτηθεί το τμήμα που χρειάζεται. Ύστερα περιμένει στον σεμαφόρο `k->sp2`. Αφού το παιδί ενημερώσει το πεδίο `k->segment` με το τμήμα που θέλει ανεβάζει τον σεμαφόρο `k->sp2` ώστε η μητρική διεργασία να αντιγράψει το τμήμα που αιτήθηκε στην μεταβλητή `str`. Το παιδί περιμένει στον σεμαφόρο `k->sp1` ώσπου η μητρική διεργασία τον ανεβάσει αφού εκείνη έχει φορτώσει το απαραίτητο τμήμα στην κοινόχρηστη μνήμη.

Όταν πια έχει φορτωθεί το τμήμα στη μνήμη και η διεργασία παιδί που το αιτήθηκε είναι έτοιμη να συνεχίσει τη λειτουργία της ελευθερώνει την επόμενη αίτηση που περιμένει στον `sp[x].sp` και η επόμενη αίτηση την παραεπόμενη κ.ο.κ. Έτσι όλες οι διεργασίες παιδιά που θέλουν γραμμή από το ίδιο τμήμα μπορούν ταυτόχρονα να διαβάζουν την γραμμή τους από το ίδιο τμήμα που είναι φορτωμένο στην κοινόχρηστη μνήμη. Κάθε διεργασία παιδί όταν τελειώνει με το «διάβασμα» της γραμμής που ήθελε ενημερώνει τον μετρητή `k->total`(αύξηση κατά 1) ο οποίος μετράει τον αριθμό των αιτήσεων που έχουν υποβληθεί από όλες τις διεργασίες παιδιά ενώ επίσης προστατεύεται από τον `k->mutex` ώστε να αποφευχθούν `race conditions`. Επίσης ενημερώνεται και ο `sp[x].num` (προστατευόμενος πάντα με τον `sp[x].mutex`) ότι δεν χρειάζεται πια αυτό το τμήμα(τον μειώνει κατά 1) και συνεχίζει στην καταχώρηση άλλης αίτησης ή στον τερματισμό του σε περίπτωση που υπέβαλε όλες τις αιτήσεις του. Αν μετά την ενημέρωση του `sp[x].num` αυτό ισούται με 0 σημαίνει ότι κανένας άλλος δεν χρειάζεται το τμήμα `x` και άρα κατεβάζουμε τον σεμαφόρο `sp[x].sp` εφόσον δεν υπάρχει επόμενο αίτημα του `x` να περάσει και ανεβάζουμε τον σεμαφόρο `k->sp2` ώστε η μητρική διεργασία να ξεκινήσει από την αρχή της `while` πάλι για την επόμενη αίτηση. Η `while` της

Κωνσταντίνος Κανελλάκης
Αριθμός Μητρώου: 1115202000064
Εργασία 1-Λειτουργικά Συστήματα

μητρικής διεργασίας τερματίζει όταν όλες οι αιτήσεις τεραμτίσουν ($k > total == N * requests$).

Τέλος η μητρική διεργασία περιμένει όλα τα παιδιά να τερματίσουν, αποσυνδέει την κοινόχρηστη μνήμη και διαγράφει το διαμοιραζόμενο τμήμα της μνήμης.