Λειτουργικά Συστήματα (Κ22) / Περίοδος 2024-2025

1η Εργασία

Το πρόγραμμα περιλαμβάνει μία διεργασία-γονέα (P) που αναλαμβάνει τη δημιουργία, τον συντονισμό και τον τερματισμό συνεργαζόμενων διεργασιών-παιδιών (C1, C2, …, Ck) σύμφωνα με οδηγίες που λαμβάνει από ένα αρχείο εντολών (Config File - CF). Ταυτόχρονα, η διεργασία P διαχειρίζεται την επικοινωνία με τα παιδιά μέσω κοινής μνήμης (shared memory) και σημαφόρων (semaphores). Η διεργασία P λαμβάνει ως είσοδο δύο αρχεία: το Config File (CF), το οποίο περιέχει οδηγίες για τις διεργασίες που πρέπει να εκτελεστούν, και ένα Text File, από το οποίο αντλούνται τυχαίες γραμμές που αποστέλλονται στις διεργασίες-παιδιά. Επιπλέον, η εφαρμογή λαμβάνει μία ακέραια τιμή M που ορίζει το μέγεθος του πίνακα των σημαφόρων καθορίζοντας έτσι και τον μέγιστο αριθμό ενεργών παιδιών ενώ δημιουργεί κοινόχρηστα αρχεία (log files) για την καταγραφή της δραστηριότητας των διεργασιών.

Οδηγίες Μεταγλώττισης

1. make clean
2. make
3. ./process_simulation <config.txt name> mobydick.txt <M >

Child.c

Το αρχείο child.c είναι ένα θεμελιώδες μέρος του συστήματος διεργασιών και έχει ως στόχο την υλοποίηση της λειτουργικότητας της κάθε διεργασίας-παιδί (child process) στο πλαίσιο της επικοινωνίας μεταξύ γονέα (parent) και παιδιών μέσω μηχανισμών IPC (Inter-Process Communication), όπως οι σημαφόροι και η διαμοιραζόμενη μνήμη. Η βασική λειτουργία της διεργασίας-παιδί είναι να βρίσκεται σε κατάσταση αναμονής (blocked) μέχρι να λάβει σήμα από τη διεργασία-γονέα. Κατά την παραλαβή ενός μηνύματος, ελέγχεται εάν το μήνυμα είναι "TERMINATE". Αν ναι, τότε η διεργασία καταγράφει το μήνυμα τερματισμού στο αρχείο καταγραφής της, υπολογίζει τη διάρκεια της ενεργής λειτουργίας της, καταγράφει το συνολικό αριθμό γραμμών που παρέλαβε από τον γονέα και στη συνέχεια τερματίζει, ειδοποιώντας ταυτόχρονα τον γονέα μέσω του σημαφόρου (parentNotificationSemaphore). Σε οποιαδήποτε άλλη περίπτωση, το παιδί καταγράφει τη γραμμή που έλαβε από τον γονέα και συνεχίζει την αναμονή για το επόμενο μήνυμα. Το πρόγραμμα ξεκινά με τη δημιουργία ή το άνοιγμα ενός μοναδικού αρχείου καταγραφής για κάθε διεργασία-παιδί, βασισμένου στο PID

της διεργασίας, εξασφαλίζοντας ότι κάθε παιδί έχει τον δικό του ξεχωριστό χώρο για να καταγράψει τις δραστηριότητές του. Η κύρια λογική της διεργασίας υλοποιείται μέσα σε έναν άπειρο βρόχο, όπου το παιδί περιμένει το σήμα από τον σημαφόρο που του έχει ανατεθεί.

Common.c

Ουσιαστικά εξυπηρετεί τις λειτουργίες της διαμοιραζόμενης μνήμης, διευκολύνοντας τη διαχείριση των διεργασιών-παιδιών, την επικοινωνία μεταξύ γονέα και παιδιών, καθώς και τη διαχείριση των κοινών πόρων, όπως η κοινόχρηστη μνήμη και οι σημαφόροι. Συγκεκριμένα, η συνάρτηση freeSlotFinder εντοπίζει την πρώτη διαθέσιμη θέση στον πίνακα των παιδιών, επιτρέποντας στον γονέα να γνωρίζει πού μπορεί να δημιουργήσει μια νέα διεργασία-παιδί. Η συνάρτηση childLabelFinder αναζητά μια συγκεκριμένη διεργασία με βάση την ετικέτα της, διευκολύνοντας τον γονέα να εντοπίσει και να διαχειριστεί συγκεκριμένες διεργασίες. Η συνάρτηση lineCount μετρά τον συνολικό αριθμό γραμμών σε ένα δοσμένο αρχείο, χρήσιμη για την επιλογή τυχαίων γραμμών. Η συνάρτηση randomActiveChildrenSelection επιλέγει τυχαία μια ενεργή διεργασία-παιδί από τον πίνακα των παιδιών, επιτρέποντας στον γονέα να στείλει δεδομένα σε μια τυχαία ενεργή διεργασία. Η συνάρτηση randomLineSelection ανακτά μια τυχαία γραμμή από ένα αρχείο, εξασφαλίζοντας ομοιόμορφη κατανομή στην επιλογή των γραμμών. Η συνάρτηση sharedMemSetup αρχικοποιεί και ρυθμίζει το τμήμα κοινόχρηστης μνήμης για την IPC, επιτρέποντας την ανταλλαγή δεδομένων μεταξύ γονέα και παιδιών. Η συνάρτηση rescourceCleanup καθαρίζει τους πόρους, όπως την κοινόχρηστη μνήμη και τους σημαφόρους, μετά τη χρήση τους, διασφαλίζοντας ότι δεν υπάρχουν διαρροές πόρων. Τέλος, η συνάρτηση semInit αρχικοποιεί έναν πίνακα σημαφόρων που χρησιμοποιούνται για τον συγχρονισμό μεταξύ γονέα και παιδιών, με κάθε σημαφόρο να έχει μοναδικό όνομα και να αρχικοποιείται στο 0. Στον ίδιο πίνακα περιλαμβάνεται και ο σημαφόρος parentNotificationSemaphore, που χρησιμοποιείται για την επικοινωνία του γονέα με όλα τα παιδιά.

Common.h

Εδώ ορίζεται η δομή που εκφράζει την διαμοιραζόμενη μνήμη, η SharedMemory. Περιλαμβάνει τα πεδία activeChildIndx, που δείχνει το index της τρέχουσας ενεργής διεργασίας, endsInTimestamp, που αποθηκεύει τη χρονική σήμανση τερματισμού για την πιο πρόσφατα τερματισμένη διεργασία, και sharedSpace, έναν buffer που χρησιμοποιείται για την ανταλλαγή μηνυμάτων, όπως γραμμές κειμένου ή μηνύματα

τερματισμού. Επιπλέον, στο common.h περιλαμβάνονται σταθερές, όπως οι MAX_CONFIG_LINES, MAX_LINE_SIZE, MAX_CHILDREN, και MAX_REQUESTS, που καθορίζουν τα όρια του συστήματος, όπως το μέγιστο πλήθος παιδιών ή η μέγιστη χωρητικότητα γραμμών σε buffers. Έβαλα αυτές τις σταθερές κυρίως για ασφάλεια. Εάν τα test cases σας απαιτούν ενδεχομένως μεγαλύτερο αριθμό από εκείνον τον οποίο εγώ όρισα, μπορείτε να αυξήσετε.

Parent.c

Το αρχείο parent.c αποτελεί την καρδιά του προγράμματος και υλοποιεί τη λογική της διεργασίας-γονέα (Parent Process), η οποία είναι υπεύθυνη για τον συντονισμό του συστήματος, τη δημιουργία, τη διαχείριση και τον τερματισμό των διεργασιών-παιδιών, καθώς και για την επικοινωνία μαζί τους μέσω μηχανισμών IPC (Inter-Process Communication). Η διεργασία-γονέας διαβάζει τις οδηγίες από ένα αρχείο (configuration file) και εκτελεί τις εντολές που σχετίζονται με τη δημιουργία (SPAWN) και τον τερματισμό (TERMINATE) των παιδιών, καθώς και την αποστολή τυχαίων γραμμών από ένα αρχείο κειμένου σε ενεργές διεργασίες-παιδιά. Η βασική λειτουργία ξεκινά με την ανάγνωση και ανάλυση του αρχείου ρυθμίσεων μέσω της συνάρτησης configInfo. Αυτή επιστρέφει έναν πίνακα δομών ConfigEntry, όπου κάθε εγγραφή περιλαμβάνει μια χρονοσήμανση, την ετικέτα της διεργασίας και την εντολή (SPAWN, TERMINATE, ή EXIT). Με βάση αυτόν τον πίνακα, ο γονέας καθορίζει ποιες ενέργειες πρέπει να εκτελεστούν σε κάθε χρονικό στιγμιότυπο. Το σύστημα χρησιμοποιεί έναν πίνακα σημαφόρων, μεγέθους M, όπου κάθε σημαφόρος αντιστοιχεί σε μία διεργασία-παιδί, για να συγχρονίζει την επικοινωνία. Επιπλέον, χρησιμοποιείται διαμοιραζόμενη μνήμη (SharedMemory) για την αποστολή δεδομένων από τον γονέα στα παιδιά. Η αρχικοποίηση αυτών των πόρων πραγματοποιείται με τις συναρτήσεις semInit και sharedMemSetup. Ο πίνακας children αποθηκεύει τα PID των ενεργών διεργασιών-παιδιών, ενώ τα childLabels και activationTime αποθηκεύουν αντίστοιχα τις ετικέτες και τις χρονικές στιγμές ενεργοποίησης των παιδιών. Η κύρια λογική υλοποιείται μέσα σε έναν βρόχο που διατρέχει τα χρονικά στιγμιότυπα από το μηδέν μέχρι τη χρονοσήμανση της εντολής EXIT. Σε κάθε χρονικό στιγμιότυπο, ο γονέας εξετάζει τις εντολές που πρέπει να εκτελεστούν. Αν η εντολή είναι SPAWN, δημιουργεί μια νέα διεργασία-παιδί με τη χρήση της fork και αναθέτει τον αντίστοιχο σημαφόρο και τις πληροφορίες. Αν η εντολή είναι TERMINATE, ο γονέας στέλνει μήνυμα τερματισμού μέσω της κοινόχρηστης μνήμης, ενεργοποιεί τον σημαφόρο για να ξεμπλοκάρει το παιδί και περιμένει επιβεβαίωση από τον parentNotificationSemaphore. Εκτός από τις εντολές SPAWN και TERMINATE, ο γονέας σε κάθε κύκλο στέλνει τυχαία επιλεγμένες γραμμές από το αρχείο κειμένου σε τυχαία ενεργές διεργασίες-παιδιά, εξασφαλίζοντας την ομοιόμορφη κατανομή των μηνυμάτων. Οι διεργασίες που σταματούν εξαιτίας της

EXIT και όχι της TERMINATE εμφανίζουν το μήνυμα «Terminated process ..... - BRUTE-FORCE (because of the exit).» Επιπλέον, όταν γίνεται προσπάθεια τερματισμού μη ενεργής διεργασίας, εμφανίζεται το μήνυμα «Terminate command issued for non-existent or inactive process».

** Κατά την εκτέλεση του προγράμματος, εάν θέλετε να προσομοιώσετε την εκτέλεση με το timestamp, μπορείτε να προσθέσετε sleep(1). Δεν το υλοποίησα για να φανει ότι ο συγχρονισμός των σημαφόρων μου και η λειτουργικότητα του προγράμματος συνολικά δεν στηρίζεται στο sleep.

**Στην κονσόλα φαίνεται μια συνολική εικόνα της όλης εκτέλεσης του προγράμματος και παρατηρούμε ότι ,όπως ζητείται, σε ένα συγκεκριμένο κύκλο λειτουργίας / timestamp ο γονέας στέλνει μια γραμμή και στον ίδιο κύκλο το παιδί την λαμβάνει, τυπώνοντάς την. Για περισσότερες πληροφορίες όσον αφορά το παιδί, θεώρησα πιο συνετό να είναι οργανωμένες σε αρχεία, .log files με το pid του εκάστοτε παιδιού, όπου φαίνονται πάλι τα μηνύματα που έχει λάβει το παιδί από τον γονέα και στο τέλος υπολογίζεται και τόσο ο χρόνος για τον οποίο έμεινε ενεργή μια διεργασία , όσο και ο αριθμος μηνυμάτων που ελήφθησαν.

**Κάνω την παραδοχή ότι όταν ένα παιδί κάνει terminate, δεν περιορίζει τον γονέα από το να στέλνει μια τυχαία γραμμη σε κάποιο αλλο ενεργό παιδί στο ίδιο κύκλο.