Progetto di sistemi operativi: Reazione a catena

Francesco Mauro, Riccardo Oro

A.A 2023/2024

Contents

1 Master

- Si occupa di leggere da file attraverso la funzione scan_data i valori di configurazione necessari per la simulazione che verranno associati alla struct config.
- Crea N_ATOMI_INIT atomi usando la funzione fork. Il processo figlio generato avrà associato un array contenente i parametri di configurazione personalizzati e il PID del processo master, eseguirà un `execvp` a `./bin/atom`.
- Crea il processo `psu` usando la funzione fork. Il processo figlio generato avrà associato un array contenente i parametri di configurazione personalizzati, il PID del processo master e gli ID di alcuni IPC, eseguirà un `execvp` a `./bin/psu`.
- Crea il processo `inhibitor` sotto richiesta dell'utente attraverso la funzione fork. Il processo figlio generato avrà associato un array contenente i parametri di configurazione personalizzati e il PID del processo master, eseguirà un `execvp` a `./bin/inhibitor`.
- Si occupa di registrare i segnali che verranno gestiti dagli appositi handler, che permettono la gestione delle terminazioni della simulazione.
- Dopo aver inizializzato tutto il necessario, imposta un alarm con valore `SIM_DURATION` che rappresenta la durata complessiva della simulazione. Il segnale `SIGALARM` è gestito attraverso un handler.

- Ogni secondo mostra a schermo:
  - Il numero di fissioni avvenute nell'ultimo secondo.
  - Il numero di scorie raccolte nell'ultimo secondo.
  - L'energia prodotta dei processi atomo nell'ultimo secondo.
  - I bilanciamenti effettuati di processo inhibitor per limitare le fissioni dei processi atomo nell'ultimo secondo.
- Si occupa in seguito allo scadere di un timer visibile a schermo di far partire i processi atomi per la simulazione.
- Si occupa di gestire le terminazioni della simulazione.

2 Terminazione

Le terminazioni della simulazione sono gestite dal processo master attraverso la funzione why_term che si occupa di interrompere la simulazione nel caso si verifichi uno dei seguenti casi ripartiti nella richiesta progettuale, ogni terminazione è identificata da una macro che sono contenute in un enum apposito. Ogni volta che si presenta una terminazione avviene:

- Rimozione degli oggetti IPC utilizzati dai vari processi durante la simulazione.
- Rimuovere attraverso il segnale `SIGKILL` tutti i processi ancora attivi per la simulazione.
- Stampare su schermo la motivazione della terminazione.

3 Atom

Il processo atomo viene generato attraverso fork dal processo master ,dopo la sua nascita fa le seguenti operazioni.

- Inizializza gli `IPC` necessari.
- Registra il segnale `SIGCHLD` associandolo all'handler apposito.
- Registra il segnale `SIGUSR1` associandolo ad un handler che si occupera in caso si verificasse `MELTDOWN` di inviare il segnale `SIGUSR1` al processo master che gestira la terminazione della simulazione.
- Preleva e associa all'apposita struct i valori della configurazione attraverso argv.

Dopo queste operazioni il processo atomo si autoinvia un segnale `SIGSTOP` per far capire che lui ha inizializzato tutto il necessario ,attendendo dal processo atomo il segnale `SIGCONT` che arriverà nel momento in cui partirà quando scadrà il timer a schermo e verrà impostato l'alarm con `SIM_DURATION`. Successivamente il processo atomo si mette all'opera e dopo aver ricevuto e prelevato il comando di fissione dalla message queue che lo mette in comunicazione con il processo attivatore ,la fissione dell'atomo è gestita dalla funzione atom_fission l'atomo saprà se effettuare o meno la fissione dopo avere prelevato il valore all'interno della messagge queue ,in caso di comando positivo :

- Nel caso in cui il suo numero atomico sarà inferiore a quello prestabilità all'interno della configurazione attraverso il parametro `MIN_A_ATOMICO` esso non potrà effettuare fissione e aumenterà il numero delle scorie.
-