UNIVERSITÀ
DI TORINO

Progetto di Sistemi Operativi 2023/2024

Reazione a catena

Ismael Oulain

Introduzione

Questo documento descrive brevemente l'implementazione del progetto di simulazione di una reazione a catena. Il progetto dimostra l'uso di processi concorrenti e la gestione delle risorse in un ambiente operativo UNIX-like, evidenziando le tecniche di sincronizzazione e comunicazione inter-processo (IPC).

Architettura del progetto

Il sistema è organizzato in quattro principali tipi di processi: **Master**, **Atomo**, **Attivatore**, e **Alimentatore**. Questi processi cooperano per simulare la reazione a catena, mirando a massimizzare la concorrenza e gestire efficientemente gli eventi all'interno della simulazione. L'architettura del progetto è stata resa modulare, suddividendo le funzionalità in file sorgente e header specifici per ogni componente. Questo approccio non solo facilita la manutenzione del codice ma permette anche una più chiara separazione delle responsabilità tra le varie parti del sistema.

Descrizione delle Componenti Chiave

Processo Master

Coordinatore della simulazione, crea i processi atomo e gestisce i processi attivatore e alimentatore

1. **Creazione delle Risorse:**
    a. **Code di Messaggi**: Una coda di messaggi viene creata per comunicare tra i vari processi, principalmente per confermare la creazione e la terminazione.
    b. **Memoria Condivisa**: Due segmenti di memoria condivisa vengono creati, uno per mantenere gli ID dei processi atomo e uno per le statistiche della simulazione.
    c. **Semafori:** Vengono creati due semafori per gestire l'accesso alle strutture condivise: uno per gli atomi e uno per le statistiche.
2. **Gestione del processo:**
    a. Inizia creando i processi **Atomo** poi successivamente **Attivatore** e **Alimentatore**
    b. Ricevuta la conferma di creazione dei processi, inizia a impostare la durata della simulazione e avvisa a tutti i processi di iniziare la simulazione tramite l'uso delle **Signal**.
    c. Avviata la simulazione rimane in ascolto di eventuali segnali di **TIMEOUT**, **EXPLODE, BLACKOUT** e **MELTDOWN**
    d. Per stampare ogni secondo le statistiche il master si avvale di un altro processo figlio che dovrà occuparsi di stampare ogni secondo e gestire l'energia lanciando eventualmente segnali di **EXPLODE** e **BLACKOUT**

Processo Atomo

Simula la scissione atomica mediante fork, essenziale per la dinamica della simulazione.

1. **Gestione del processo:**
    a. Alla sua creazione il processo Atomo recupera le informazioni sulle memorie condivise e i semafori attraverso le variabili di ambiente e successivamente informa tramite la coda di messaggi la sua creazione al processo Master
    b. Successivamente salva il suo pid nella struttura dati e rimane in attesa del **segnale di scissione** dall'Attivatore sempre utilizzando le **Signal.**
    c. Alla ricezione del segnale di scissione si attiva l'handler verifica se il numero atomico permette la scissione:
        i. Se lo permette effetua una fork creando un nuovo atomo aumenta le statistiche delle scissioni e dell'energia.
        ii. Altrimenti diventa una scoria, elimina il suo pid dal registro ed aumenta le statistiche delle scorie.

Processo Attivatore

L'attivatore invia segnali di scissione agli atomi selezionati a intervalli regolari, orchestrando parte della logica di simulazione attraverso segnali e semafori.

1. **Gestione del processo:**
    a. Alla sua creazione informa il processo Master della sua creazione tramite la coda di messaggi e recupera le informazioni relative alle memorie e semafori tramite le variabili di ambiente
    b. Rimane in attesa del segnale di "Avvio Simulazione" tramite la **Signal**(SIGUSR1).
    c. All'avvio della simulazione attiva il timer e ogni **STEP_ATTIVATORE** invia il segnale (SIGUSR1) di scissione selezionando RANDOM_ATOM_SET atomi dal registro.
    d. Aggiorna le statistiche.

Processo Alimentatore

L'alimentatore aggiunge nuovi atomi nella simulazione, simulando l'introduzione di nuovo combustibile. Utilizza timer per gestire la creazione periodica di nuovi processi atomo.

1. **Gestione del processo:**
    a. Alla sua creazione informa il processo Master della sua creazione tramite la coda di messaggi e recupera le informazioni relative alle memorie e semafori tramite le variabili di ambiente
    b. Ogni STEP_ATTIVATORE crea N_NUOVI_ATOMI eseguendo una fork con il codice del processo Atomo

Guida alla Compilazione ed Esecuzione

Settaggio dei Parametri

Modificare i parametri di configurazione nel file **utils.h** per adeguare la simulazione alle proprie esigenze. Qui si può impostare variabili come la durata della simulazione, il numero di atomi iniziali, ecc.

Compilazione

Per compilare il progetto:

```
make all
```

Questo comando costruisce tutti gli eseguibili necessari per il progetto.

Pulizia

Per rimuovere gli eseguibili e pulire il progetto:

```
make clean
```

Esecuzione

Per avviare la simulazione:

Make run oppure **./master**