Sistemi Operativi

Compito 03-02-2025

Esercizio 1 (10 punti)

Un sistema operativo adotta la politica di scheduling dei thread a coda multipla con prelazione tra le code. Sono presenti tre code, una ad alta priorità (H) con scheduling round-robin e quanto q=2ms, una a media priorità (M) con scheduling round-robin e quanto q=3ms e una a bassa priorità (L) con scheduling FCFS. Un thread nella coda M che usa tutto il quanto viene spostato nella coda L mentre se esegue tutto il CPU burst entro 1.5ms viene promosso alla coda H. Un thread della coda H che usa tutto il quanto di tempo viene spostato nella coda M. Un thread della coda L al termine del I/O viene inserito nella coda M.

Quando un thread viene prelazionato viene reinserito in fondo alla coda e quando verrà rieseguito riceverà un nuovo quanto di tempo.

Il sistema deve schedulare i seguenti thread con tempi di arrivo, coda e uso CPU/IO indicati:

T_1 T_{arrivo}=2 coda M CPU(2ms)/IO(6ms)/CPU(2ms)/IO(5ms)/CPU(3ms)

T_2 T_{arrivo}=1 coda M CPU(1ms)/IO(6ms)/CPU(1ms)/IO(6ms)/CPU(1ms)

T_3 T_{arrivo}=0 coda M CPU(5ms)/IO(5ms)/CPU(4ms)

T_4 T_{arrivo}=3 coda M CPU(5ms)/IO(5ms)/CPU(2ms)

Si determini: il **diagramma di Gantt**, il **tempo di attesa** medio, il **tempo di ritorno** medio, il **tempo di risposta** medio, il numero di **cambi di contesto** ed indicare gli istanti in cui si ha prelazione tra le code. Nel diagramma di Gantt indicare in quale coda avviene l'esecuzione del thread.

Esercizio 2 (20 punti)

Si vuole realizzare in java il seguente sistema:

Sono presenti N *WriterThread* che iterativamente producono un array di K valori interi impiegando un tempo variabile in [TG, TG+DG), ogni thread deve sommare l'array generato su un unico array condiviso in modo che la somma avvenga in modo atomico (per evitare race conditions). E' presente un thread *SamplerThread* che iterativamente legge in modo atomico l'array condiviso (deve impedire la scrittura durante la lettura) e inserisce in una coda illimitata l'array insieme ad un numero progressivo ed aspetta T ms. Definire la classe *SampleData* per tenere il numero progressivo e l'array dei K numeri.

Sono presenti M *ProcessorThread* dove ognuno iterativamente deve estrarre 2 oggetti *SampleData* con due array consecutivi (deve aspettare se entrambi non sono presenti) calcolare la differenza dei due array (il secondo meno il primo) e quindi inserire l'array con il progressivo del primo array in una coda limitata Q. Sono infine presenti due *PrinterThread* dove ognuno iterativamente estrae un array dalla coda Q e stampa su una riga il progressivo e l'array estratto.

Per facilitare il debugging i *WriterThread* generano array con tutti i K valori uguali a 1.

Il programma principale fa partire i thread quindi dopo 10 secondi vengono interrotti tutti i thread e deve stampare per ogni *WriterThread* il numero di array sommati, per il *SamplerThread* e ogni *ProcessorThread e PrinterThread* il numero di operazioni completate ed il relativo totale ed infine il numero di oggetti rimasti nelle code.

Realizzare il sistema usando i **semafori** per la sincronizzazione dei thread (usare attesa attiva o polling sarà considerato errore).