

Elementi di Sistemi Operativi

Bioinformatica - Tiziano Villa

7 Luglio 2008

Nome e Cognome:

Matricola:

Posta elettronica:

problema	punti massimi	i tuoi punti
problema 1	8	
problema 2	8	
problema 3	8	
problema 4	6	
totale	30	

1. Rispondere in modo preciso ma conciso alle seguenti domande.

- (a) Si descrivano succintamente i servizi di un sistema operativo relativamente alle operazioni di ingresso/uscita e alla gestione del "file system".

Traccia di soluzione.

Si veda la Sez. 2.1 del testo di Silberschatz.

- (b) Si descrivano succintamente i servizi di un sistema operativo relativamente alla protezione e sicurezza.

Traccia di soluzione.

Si veda la Sez. 2.1 del testo di Silberschatz.

- (c) Quali sono i vantaggi e gli svantaggi derivanti dall'usare la medesima interfaccia di chiamate a sistema per gestire sia "files" che dispositivi d'ingresso/uscita ?

Traccia di soluzione.

Si puo' accedere a ogni dispositivo come ad un "file". Vantaggio: dato che il sistema operativo si relaziona ai dispositivi attraverso questa interfaccia, e' facile aggiungere un nuovo dispositivo scrivendone il codice specifico chiamato attraverso questa interfaccia astratta. Questo semplifica sia la scrittura del codice utente che puo' accedere a dispositivi e "files" nella medesima maniera, che il codice dei piloti dei dispositivi ("device driver") che puo' fare riferimento a un API ben definito.

Lo svantaggio nell'usare la medesima interfaccia e' che potrebbe essere difficile definire la funzionalita' di certi dispositivi usando l'API per accedere ai files, risultando in perdita' di funzionalita' o prestazione. Per alleviare il problema si potrebbe usare l'operazione `ioctl` che mette a disposizione un'interfaccia generale attraverso cui i processi possono invocare invocare operazioni sui dispositivi.

(d) Si descrivano i modi per gli utenti di comunicare con il sistema operativo.

Traccia di soluzione.

Si veda le Sez. 2.1 e 2.2 del testo di Silberschatz.

2. Si consideri il seguente codice per risolvere il problema dei produttori e consumatori con memoria limitata mediante semafori.

Inizialmente ci sono `numBuffers` vuoti e nessun pieno. Se non ci sono vuoti, il produttore deve aspettare che il consumatore prelevi qualche elemento, trasformando dei pieni in vuoti. Se non ci sono pieni, il consumatore deve aspettare che il produttore aggiunga qualche elemento, trasformando dei vuoti in pieni. Soltanto un processo per volta puo' operare sulla struttura condivisa. Il sistema non deve entrare mai in stallo.

```
Semaphore fullBuffers = 0;
Semaphore emptyBuffers = numBuffers;
Semaphore mutex = 1;
```

```
Producer(item) {
    mutex.P();
    emptyBuffers.P();
    Enqueue(item);
    mutex.V();
    fullBuffers.V();
}
```

```
Consumer() {
    fullBuffers.P();
    mutex.P();
    item = Dequeue();
    mutex.V();
    emptyBuffers.V();
    return item;
}
```

- (a) Si analizzi il codice precedente per stabilire se realizza correttamente le specifiche, spiegandone il funzionamento e correggendo eventuali errori. Traccia di soluzione.

L'errore consiste nell'inversione delle due operazioni P iniziali di `Producer`. Infatti il codice proposto può provocare uno stallo, poiché potrebbe succedere che non ci siano vuoti, il produttore entri nella sezione critica e si blocchi su `emptyBuffers.P()`, ma il consumatore non possa prelevare alcun elemento perché escluso dalla sezione critica bloccata dal produttore a causa del suo `mutex.P()`.

La soluzione corretta segue.

```
Semaphore fullBuffers = 0;
Semaphore emptyBuffers = numBuffers;
Semaphore mutex = 1;
```

```
Producer(item) {
    emptyBuffers.P();
    mutex.P();
    Enqueue(item);
    mutex.V();
    fullBuffers.V();
}
```

```
Consumer() {
    fullBuffers.P();
    mutex.P();
    item = Dequeue();
    mutex.V();
    emptyBuffers.V();
    return item;
}
```

- (b) Se si scambia l'ordine delle due operazioni V finali di `Producer` il codice e' ancora corretto ?

Traccia di soluzione.

Si. L'unico effetto e' che puo' diminuire l'efficienza della schedulazione.

Si consideri la seguente successione di eventi:

```
produttore: fullBuffers.V() poi perde il processore
consumatore: fullBuffers.P()
              mutex.P() poi si blocca (*)
              ((*) manca mutex.V del produttore))
produttore: mutex.V    poi finisce
consumatore: mutex.P    e prosegue
```

che mostra come il consumatore potrebbe essere temporaneamente bloccato in `mutex.P()` in attesa che il produttore esegua il suo `mutex.V()` (scambiato per ipotesi rispetto a `fullBuffers.V()`). Percio' il processore dovrebbe essere assegnato di nuovo al produttore perche' possa eseguire il suo `mutex.V()`. Quindi si avrebbero per i processi piu' cambi di contesto del minimo indispensabile, diminuendo l'efficienza.

(c) La soluzione proposta vale anche per 2 produttori o 2 consumatori ?

Traccia di soluzione.

Si.

3. Siano dati 5 processi con la durata della sequenza di operazioni della CPU espressa in millesecondi.

Processo	Durata	Priorita'
P1	10	3
P2	1	1
P3	2	3
P4	1	4
P5	5	2

Si supponga che i processi siano arrivati nell'ordine $P1, P2, P3, P4, P5$ e che siano tutti presenti al tempo 0.

- (a) i. Si disegni lo schema GANTT (come nel libro di testo) che illustri l'esecuzione di questi processi con l'algoritmo di schedulazione con priorita' senza prelazione (un numero di priorita' piu' basso indica una priorita' maggiore).

Traccia di soluzione.

P2	P5	P5	P5	P5	P5	P1	P1	P1	P1	P1	P1	P1	P1	P1	P1	P1	P3	P3	P4
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

- ii. Si calcoli il tempo di completamento di ciascun processo.

Traccia di soluzione.

I tempi di completamento sono:

P1 16,

P2 1,

P3 18,

P4 19,

P5 6

- iii. Si calcoli il tempo di attesa di ciascun processo.

Traccia di soluzione.

I tempi di attesa sono (tempo di completamento - durata):

P1 6,

P2 0,

P3 16,

P4 18,

P5 1

- iv. Si calcoli il tempo di attesa medio tra tutti i processi.

Traccia di soluzione.

Tempo di attesa medio: $(6+0+16+18+1)/5 = 8,2$ ms.

- (b) i. Si disegni lo schema GANTT (come nel libro di testo) che illustri l'esecuzione di questi processi con l'algoritmo di schedulazione RR (Round Robin) con quanto temporale = 1.

Traccia di soluzione.

P1	P2	P3	P4	P5	P1	P3	P5	P1	P5	P1	P5	P1	P5	P1	P1	P1	P1	P1	
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

- ii. Si calcoli il tempo di completamento di ciascun processo.

Traccia di soluzione.

I tempi di completamento sono:

P1 19,

P2 2,

P3 7,

P4 4,

P5 14

- iii. Si calcoli il tempo di attesa di ciascun processo.

Traccia di soluzione.

I tempi di attesa sono (tempo di completamento - durata):

P1 9,

P2 1,

P3 5,

P4 3,

P5 9

- iv. Si calcoli il tempo di attesa medio tra tutti i processi.

Traccia di soluzione.

Tempo di attesa medio: $(9+1+5+3+9)/5 = 5,4$ ms.

4. L'interazione con le unita' d'ingresso/uscita puo' avvenire tramite interrogazione ciclica ("polling") o interruzioni ("interrupt"). Si vuole studiare la convenienza di usare il sistema dell'interrogazione per gestire le unita' d'ingresso/uscita.

Si supponga che il numero di cicli di orologio per un'interrogazione sia pari a 400, includendo il trasferimento del controllo alla procedura d'interrogazione, l'accesso al dispositivo ed il ritorno al programma utente. Inoltre si supponga che il processore lavori con una frequenza di 500 MHz.

Nell'ipotesi che si esegua un'interrogazione del dispositivo d'ingresso/uscita con una frequenza sufficiente affinche' nessun dato sia mai perso, e che il dispositivo sia sempre al lavoro, si determini la frequenza di tempo del processore consumato dal meccanismo dell'interrogazione per i seguenti casi.

- (a) Il dispositivo e' il "mouse"; si supponga che sia interrogato 30 volte al secondo per garantire di non perdere alcun movimento fatto dall'utente.

E' conveniente l'interrogazione in questo caso ?

Traccia di soluzione

I cicli di orologio per l'interrogazione al secondo sono $30 \times 400 = 12000 = 12 \times 10^3$ cicli al secondo.

In un secondo il processore ha a disposizione 500×10^6 cicli (questo e' il significato di 500 MHz al secondo). Percio' la frazione di cicli utilizzati e' data da $12 \times 10^3 / 500 \times 10^6 = 0,0024\%$.

Si puo' concludere che in questo caso e' conveniente usare il meccanismo dell'interrogazione per il basso impatto sulla prestazione del processore.

- (b) Il dispositivo e' il disco rigido; si supponga che esso trasferisca dati in blocchi di 4 parole e che possa eseguire il trasferimento alla velocita' di 4 MB/s.

E' conveniente l'interrogazione in questo caso ?

Traccia.

E' necessario eseguire l'interrogazione con una frequenza pari alla frequenza dei blocchi di 4 parole, che e' uguale a 250 K volte al secondo (4 MB/secondo / 16 B per trasferimento = 250 K trasferimenti /secondo - si noti che 4 parole = 16 B).

Traccia di soluzione

I cicli di orologio per l'interrogazione al secondo sono $250K \times 400 = 100 \times 10^3 K$ cicli al secondo.

Ignorando la discrepanza tra le unita' di misura relative alle basi e quindi assumendo che sia $1K = 1000$ (in realta' $1K = 1024$), la frazione di cicli del processore utilizzati dall'interrogazione e' data da

$$100 \times 10^6 / 500 \times 10^6 = 20\%.$$

Poiche' un quinto del processore sarebbe utilizzato solo per interrogare il disco, in questo caso non sarebbe conveniente usare il meccanismo dell'interrogazione.