UNIVERSITÀ DEGLI STUDI DI BERGAMO

Corso di laurea in ingegneria informatica
Esame di sistemi operativi – 4 febbraio 2009 SOLUZIONI

Cognome_____Nome_____Matricola_____

1.
Illustrare con precisione il problema dei lettori e scrittori e descrivere in maniera esauriente una possibile soluzione.

Cfr. capitolo 7 del libro di testo.

2.
Siano P1 e P2 due processi da eseguire periodicamente su un sistema informatico real-time con le seguenti caratteristiche:

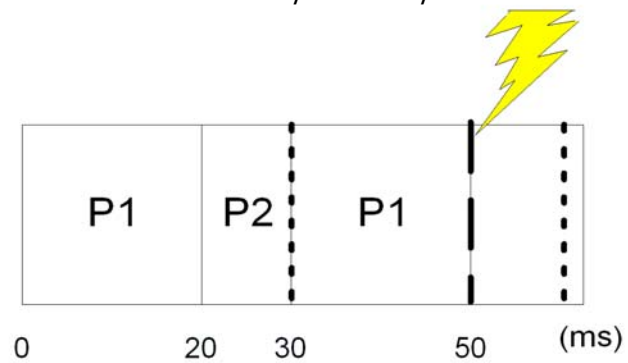
- P1 ha periodo $\pi 1$ = 30ms e tempo di esecuzione t1 = 20ms, ossia deve essere eseguito completamente una volta ogni 30ms, e un'esecuzione completa impegna il processore per 20ms;
- P2 ha periodo $\pi 2$ = 50ms e tempo di esecuzione t2 = 15ms.

a) Dimostrare che i due processi possono coesistere su un sistema a singola CPU perché il loro funzionamento combinato non supera il 100% della risorsa processore.

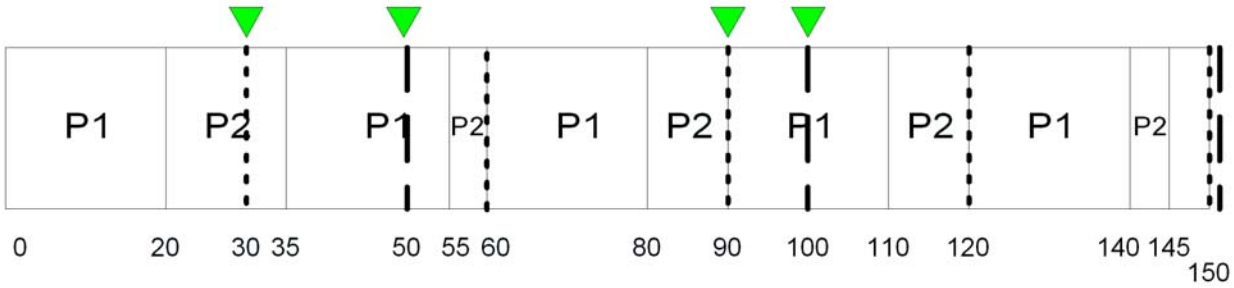
CPU%1 = 20/30 = 66,67%
CPU%2 = 15/50 = 30%
USO CPU % = CPU%1 + CPU%2 = 96,67% < 100%, quindi i due processi possono coesistere su un unico processore.

b) Mostrare su un diagramma temporale che una schedulazione a frequenza monotona (a priorità statiche inversamente proporzionali al periodo, con preemption) non è una soluzione adeguata.

A priorità statiche P1 ha sempre la precedenza in quanto con periodo più breve. A 30ms P2 non ha ancora terminato ma siamo entrati nel secondo periodo di P1 che quindi deve entrare in esecuzione causando una preemption. Non riusciamo a completare P2 prima della scadenza del suo primo periodo a 50ms.



c) Mostrare su un diagramma temporale la soluzione fornita da una schedulazione con strategia Earliest Deadline First (a priorità dinamiche, con preemption).



Con le priorità dinamiche ha precedenza il processo con la scadenza più vicina. I cambi di priorità sono indicati dai triangoli verdi. Attenzione: un cambio di priorità non necessariamente coincide con una preemption: ad esempio a 30ms, dopo la fine del primo periodo di P1, P2 acquisisce la priorità maggiore, ma essendo già in esecuzione non c'è preemption. P2 era già in esecuzione in quanto il processo a priorità maggiore (fino a 30ms), cioé P1, aveva già completato la sua esecuzione per il primo periodo.

3.
Si consideri il programma seguente:

```
1.   int main () {
2.    int i, j, pid, stat;
3.      i = 10;
4.      j = 20;
5.      j--;
6.      pid = fork ();
7.      if (pid == 0) {
8.        i = i + 13;
9.        pid = fork ();
10.       if (pid != 0) {
11.         pid = wait (&stat);
12.         j = j / 2;
13.         exit (0);
14.       } else {
15.         j = j * 4;
16.         exit (0);
17.       } /* end if */
18.     } else {
19.       i++;
20.     } /* end if */
21.     return (0);
22.   } /* end main */
```

Si chiede di completare le tabelle 1, 2 e 3 riportate sotto, indicando, negli istanti di tempo specificati, il valore delle variabili i, j e pid. Attenzione:

- nel caso in cui al momento indicato la variabile non esista (in quanto non esiste il processo), riportare NE;
- quando non si può dire con certezza se la variabile esista e/o quale ne sia il valore, riportare U;
- si suppone che tutte le chiamate ai servizi di sistema abbiano sempre successo, che il sistema operativo assegni ai processi creati dei pid consecutivi, e che il processo padre abbia pid = 501;
- la frase “prima dell’istruzione X” si riferisce all’istante in cui si inizia l’esecuzione dell’istruzione X stessa.

Tabella 1. Valore delle variabili nel processo padre.	i	j	pid
Prima dell’istruzione 7	10	19	502
Prima dell’istruzione 13	U	U	U
Prima dell’istruzione 21	11	19	502

Tabella 2. Valore delle variabili nel primo processo figlio.	i	j	pid
Prima dell’istruzione 7	10	19	0
Prima dell’istruzione 13	23	9	503
Prima dell’istruzione 21	U	U	U

Tabella 3. Valore delle variabili nel primo processo nipote.	i	j	pid
Prima dell’istruzione 7	NE	NE	NE
Prima dell’istruzione 13	NE	NE	NE
Prima dell’istruzione 21	U	U	U