Elementi di Architettura e Sistemi Operativi

Bioinformatica - Tiziano Villa

27 Febbraio 2019

Nome e Cognome:
Matricola:
Posta elettronica:

problema	punti massimi	i tuoi punti
problema 1	6	
problema 2	9	
problema 3	5	
problema 4	10	
totale	30	

1. Si consideri il seguente programma che crea un processo invocando la chiamata di sistema `fork`.

```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>

#define SIZE  5

int nums[SIZE] = {0,1,2,3,4};

int main()
{
  int i;
  pid_t pid;

  pid = fork();

  if (pid == 0) {
    for (i = 0; i < SIZE; i++) {
       nums[i] *= -i;
       printf("CHILD: %d ",nums[i]); /* LINEA X */
    }
  }
  else if (pid > 0) {
    wait(NULL);
    for (i = 0; i < SIZE; i++)
       printf("PARENT: %d ",nums[i]); /* LINEA Y */
  }

  return 0;
}
```

(a) Si spieghi la semantica della chiamata di sistema `fork`.

Si analizzi il codice precedente spiegandone il funzionamento.

Traccia di soluzione.

La chiamata di sistema fork() crea un nuovo processo figlio che avra' una copia dello spazio degl'indirizzi del processo genitore. Entrambi i processi genitore e figlio continuano l'esecuzione dall'istruzione successiva alla chiamata di sistema fork(), con una differenza: la chiamata di sistema fork() restituisce il valore 0 nel processo figlio, ma restituisce l'identificatore del processo figlio (il PID diverso da 0) nel processo genitore. Il processo genitore invocando la chiamata di sistema wait() si autorimuove dalla coda dei processi pronti fino alla terminazione del figlio. Quando il processo figlio termina, il processo genitore chiude la chiamata di sistema wait() e continua con il resto del codice.

(b) Che cosa produce il programma in uscita alle LINEE X e Y ?

Traccia di soluzione.

L'uscita alla LINEA X e' CHILD: 0, CHILD: -1, CHILD: -4, CHILD: -9, CHILD: -16.

L'uscita alla LINEA Y e' PARENT: 0, PARENT: 1, PARENT: 2, PARENT: 3, PARENT: 4.

2. (a) Si spieghi l'algoritmo del banchiere, a che cosa serve e come funziona.
   Traccia di soluzione.
   Si veda il libro di testo.

(b) Si consideri la seguente situazione istantanea di un sistema

	Allocazione				Massimo			
	A	B	C	D	A	B	C	D
P0	3	0	1	4	5	1	1	7
P1	2	2	1	0	3	2	1	1
P2	3	1	2	1	3	3	2	1
P3	0	5	1	0	4	6	1	2
P4	4	2	1	2	6	3	2	5

Disponibile			
A	B	C	D
0	3	0	1

Applicando l'algoritmo del banchiere si verifichi se il sistema e' in uno stato sicuro. Si mostrino tutti i passi dell'algoritmo. Se lo stato e' sicuro, si mostri l'ordine in cui i processi possono essere completati; se lo stato non e' sicuro, si spieghi perche' non e' sicuro.
Traccia di soluzione.
=== PRIMO CICLO ===

	Allocazione				Massimo				Fabbisogno= Massimo-Allocazione			
	A	B	C	D	A	B	C	D	A	B	C	D
P0	3	0	1	4	5	1	1	7	2	1	0	3
P1	2	2	1	0	3	2	1	1	1	0	0	1
P2	3	1	2	1	3	3	2	1	0	2	0	0
P3	0	5	1	0	4	6	1	2	4	1	0	2
P4	4	2	1	2	6	3	2	5	2	1	1	3

Disponibile			
A	B	C	D
0	3	0	1

Il Fabbisogno di P0 non puo' essere soddisfatto con il Disponibile.
Il Fabbisogno di P1 non puo' essere soddisfatto con il Disponibile.
Il Fabbisogno di P2 puo' essere soddisfatto con il Disponibile (Fabbisogno di P2 $\leq$ Disponibile) e P2 puo' essere completato. Al completamento, P2 rilascera' l' Allocazione di P2 al Disponibile che diventera'

```
Disponibile
A B C D
0 3 0 1 +
3 1 2 1
-------
3 4 2 2
```

Il Fabbisogno di P3 non puo' essere soddisfatto con il Disponibile.

Il Fabbisogno di P4 non puo' essere soddisfatto con il Disponibile.

=== SECONDO CICLO ===

Il Fabbisogno di P0 non puo' essere soddisfatto con il Disponibile.

Il Fabbisogno di P1 puo' essere soddisfatto con il Disponibile e P1 puo' essere completato. Al completamento, P1 rilascera' l' Allocazione di P1 al Disponibile che diventera'

```
Disponibile
A B C D
3 4 2 2 +
2 2 1 0
-------
5 6 3 2
```

Il Fabbisogno di P3 puo' essere soddisfatto con il Disponibile e P1 puo' essere completato. Al completamento, P3 rilascera' l' Allocazione di P3 al Disponibile che diventera'

```
Disponibile
A B C D
5 6 3 2 +
0 5 1 0
-------
5 11 4 2
```

Il Fabbisogno di P4 non puo' essere soddisfatto con il Disponibile.

=== TERZO CICLO ===

Non si puo' procedere perche' P0 e P4 abbisognano di 3 unita' della risorsa D di cui ne sono disponibili solo 2 unita', percio' lo stato non e' sicuro.

3. Si consideri il seguente codice LC-3

```
OpX   JSR POP
      ADD R5, R5, #0
      BRp Exit
      NOT R0, R0
      ADD R0, R0, #1
      JSR PUSH
Exit  RET
```

Si spieghi il suo funzionamento, sia commentando le singole istruzioni che la procedura complessiva.

Traccia di soluzione

Sfila la cima della pila, ne calcola il complemento a 2 e lo rinfila sulla cima della pila.

Si noti che:

- R5=1 se la chiamata di POP non e' andata a buon fine (e quindi salta via a Exit), altrimenti R5=0.
- Il dato tolto o aggiunto da POP e PUSH si trova nel registro R0.

4. Si progetti un circuito sequenziale che riceve in ingresso una variabile binaria I e produce in uscita la variabile binaria U che riproduce fedelmente l'ingresso tranne che per ogni sequenza di 1 trasforma il primo 1 in uno 0.

   (a) Si progetti la macchina a stati finiti che modella la specifica (**tipo Moore**) disegnandone il grafo delle transizioni.

   (b) Si minimizzi la macchina a stati finiti.

   (c) Si scriva la tavola delle transizioni e la si codifichi.

   (d) Si scrivano le equazioni minimizzate della logica che genera lo stato futuro e le uscite (si mostrino le mappe di Karnaugh).

   (e) Si disegni lo schematico del circuito sequenziale con porte logiche NAND e bistabili di tipo D.

   Traccia di soluzione.

```
0 s0 s0 0
1 s0 s1 0
0 s1 s0 0
1 s1 s11 0
0 s11 s0 1
1 s11 s11 1
```

   Si noti che la MSF di Moore risponde all'ingresso con un ciclo di ritardo. Ad es., nello stato iniziale produce 0 quale che sia l'ingresso, per cui se l'ingresso inizia come 11... si produrra' 001... cioe' il primo 0 in uscita e' l'uscita iniziale posta a 0 che non dipende dal primo ingresso, la seconda uscita e' il risultato dell'azzeramento del primo 1 in ingresso, e l'1 nella terza posizione e' il risultato del mantenimento del secondo 1 dell'ingresso, etc.

   La MSF non e' minimizzabile come si verifica applicando un algoritmo di minimizzazione degli stati.