

# Elementi di Architettura e Sistemi Operativi

Bioinformatica - Tiziano Villa

20 Giugno 2019

Nome e Cognome:

Matricola:

Posta elettronica:

problema	punti massimi	i tuoi punti
problema 1	8	
problema 2	7	
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>

int main()
{
    pid_t pid, pid1;

    pid = fork();

    if (pid < 0) {
        fprintf(stderr, "Fork fallita");
        return 1;
    }
    else if (pid == 0) {
        pid1 = getpid();
        printf("figlio: pid = %d", pid);          /* A */
        printf("figlio: pid1 = %d", pid1);        /* B */
    }
    else {
        pid1 = getpid();
        printf("genitore: pid = %d", pid);        /* C */
        printf("genitore: pid1 = %d", pid1);      /* D */
        wait(NULL);
    }

    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 avrà una copia dello spazio degli 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) Si scriva che cosa si stampa alle linee A, B, C, D. Si assuma che i *pid* del genitore e del figlio siano rispettivamente 2600 e 2603. Si spieghi il ragionamento che porta alla vostra conclusione.

Traccia di soluzione.

L'uscita alla LINEA A e' *figlio* : *pid* = 0.

L'uscita alla LINEA B e' *figlio* : *pid1* = 2603.

L'uscita alla LINEA C e' *genitore* : *pid* = 2603.

L'uscita alla LINEA D e' *genitore* : *pid1* = 2600.

Si noti che *pid1* ottenuto da *getpid* e' il pid vero del processo, mentre *pid* ottenuto da *fork* ne segue la semantica, spiegata prima.

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
1	0	0	2

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
1	0	0	2

Il Fabbisogno di P0 non puo' essere soddisfatto con il Disponibile (Fabbisogno di P0  $\not\leq$  Disponibile)

Il Fabbisogno di P1 puo' essere soddisfatto con il Disponibile (Fabbisogno di P1  $\leq$  Disponibile) e P1 puo' essere completato. Al completamento, P1 rilascerà l'Allocazione di P1 al Disponibile che diventerà

Disponibile

A	B	C	D
1	0	0	2
2	2	1	0
-----			
3	2	1	2

Il Fabbisogno di P2 puo' essere soddisfatto con il Disponibile (Fabbisogno di  $P2 \leq$  Disponibile) e P2 puo' essere completato. Al completamento, P2 rilascerà l' Allocazione di P2 al Disponibile che diventerà

Disponibile

A	B	C	D
3	2	1	2
3	1	2	1
-----			
6	3	3	3

Il Fabbisogno di P3 puo' essere soddisfatto con il Disponibile (Fabbisogno di  $P3 \leq$  Disponibile) e P3 puo' essere completato. Al completamento, P3 rilascerà l' Allocazione di P3 al Disponibile che diventerà

Disponibile

A	B	C	D
6	3	3	3
0	5	1	0
-----			
6	8	4	3

Il Fabbisogno di P4 puo' essere soddisfatto con il Disponibile (Fabbisogno di  $P4 \leq$  Disponibile) e P4 puo' essere completato. Al completamento, P4 rilascerà l' Allocazione di P4 al Disponibile che diventerà

Disponibile

A	B	C	D
6	8	4	3
4	2	1	2
-----			
10	10	5	5

=== SECONDO CICLO ===

Il Fabbisogno di P0 puo' essere soddisfatto con il Disponibile (Fabbisogno di P0  $\leq$  Disponibile) e P0 puo' essere completato. Al completamento, P0 rilascerà l' Allocazione di P0 al Disponibile che diventerà

Disponibile				
A	B	C	D	
10	10	5	5	+
2	0	1	4	
-----				
13	10	6	9	

Perciò il sistema è in uno stato sicuro.



3. Si consideri il seguente codice LC-3

```
LDR R0, R5, #0
ADD R0, R0, #-2
BRnp SALTA

AND R0, R0, #0
ADD R0, R0, #5
STR R0, R5, #-1
SALTA ...
```

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

Traccia di soluzione

Si stia attenti alla semantica di LDR e STR.

LDR legge il valore in memoria all'indirizzo contenuto in R5 e lo salva in R0:  $R0 \leftarrow mem[R5]$ .

STR scrive in memoria il contenuto di R0 all'indirizzo precedente di 1 quello contenuto in R5:  $mem[R5 - 1] \leftarrow R0$ .

```
LDR R0, R5, #0    # R0 <--- mem[R5]
ADD R0, R0, #-2   # sottrae 2 da R0
BRnp SALTA        # se R0 != 2 salta

AND R0, R0, #0    # azzera R0
ADD R0, R0, #5    # somma 5 a R0
STR R0, R5, #-1   # mem[R5-1] <--- R0=5

SALTA ...
```

Se chiamiamo rispettivamente  $x$  e  $y$  le variabili cui puntano  $R5$  e  $R5-1$  (cioè  $x$  stia per  $mem[R5]$  e  $y$  per  $mem[R5-1]$ ), allora possiamo dire che il frammento di codice LC-3 precedente è la compilazione del seguente frammento di codice C:

```
if (x == 2)
    y = 5;
```

4. Si progetti un circuito sequenziale che funziona come contatore a 3 cifre binarie secondo la sequenza seguente:  $000 \rightarrow 010 \rightarrow 011 \rightarrow 101 \rightarrow 110 \rightarrow 000$ .
- (a) Si disegni il grafo delle transizioni di una macchina a stati finiti che corrisponde alla specifica. S'indichi lo stato iniziale.
- Si minimizzi il numero degli stati della macchina proposta.
- Traccia di soluzione.
- Si veda la soluzione nelle dispense.

- (b) Si scriva la tavola delle transizioni con gli stati futuri e le uscite e la si codifichi.

- (c) Supponendo di usare bistabili di tipo D, si derivino le equazioni minimizzate di eccitazione degl'ingressi dei bistabili e le equazioni minimizzate delle uscite.

- (d) Si realizzi il circuito sequenziale corrispondente con bistabili di tipo D campionati sul fronte di salita, invertitori e porte NAND (a 2, 3, o 4 ingressi). Si etichettino con chiarezza i segnali.