

Elementi di Architettura e Sistemi Operativi

Bioinformatica - Tiziano Villa

19 Luglio 2011

Nome e Cognome:

Matricola:

Posta elettronica:

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

1. Si consideri il seguente programma scritto utilizzando le API Pthreads.

```
#include <pthread.h>
#include <stdio.h>

int value = 0;
void *runner(void *param): /* il thread */

int main(int argc, char *argv[])
{
    int pid;
    pthread_t tid;
    pthread_attr_t attr;

    pid = fork();

    if (pid == 0) { /* processo figlio */
        pthread_attr_init(&attr);
        pthread_create(&tid, &attr, runner, NULL);
        pthread_join(tid, NULL);
        printf("CHILD: value = %d", value); /* RIGA C */
    }
    else if (pid > 0) { /* processo genitore */
        wait(NULL);
        printf("PARENT: value = %d", value); /* RIGA P */
    }
}

void *runner(void *param) {
    value = 5;
    pthread_exit(0);
}
```

(a) 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.

L'esecuzione del processo figlio chiama delle procedure di tipo Pthreads. La dichiarazione di variabili `pthread_t tid` specifica l'identificatore per il thread da creare. Ogni thread ha un insieme di attributi che includono la dimensione della pila e informazioni sulla schedulazione. La dichiarazione `pthread_attr_t attr` riguarda la struttura dati per gli attributi del thread, i cui valori si assegnano con la chiamata di funzione `pthread_attr_init(&attr)`. La chiamata di funzione `pthread_create()` crea un nuovo thread. Il primo e il secondo thread condividono il valore globale di `value`. Oltre all'argomento identificatore del thread e dei suoi attributi, si passa anche il nome della funzione da cui il nuovo thread inizierà l'esecuzione (in questo caso la funzione `runner()`), e il suo parametro (in questo caso nessun parametro). Quindi nel nostro caso il processo figlio ha un thread iniziale (o genitore) che parte da `main()` e un thread figlio da lui creato che esegue la funzione `runner()` la quale cambia il valore di `value`. Dopo aver creato il secondo, il primo thread ne attende il completamento chiamando la funzione `pthread_join()`. Il secondo thread termina quando si esegue la funzione `pthread_exit()`, dopo di che il primo thread stampa in uscita il valore di `value` condiviso tra il primo e il secondo thread.

(b) Quale sarà il valore stampato alla RIGA C ?

Quale sarà il valore stampato alla RIGA P ?

Spiegare le vostre risposte.

Traccia di soluzione.

Il valore stampato alla RIGA C è 5.

Il valore stampato alla RIGA P è 0.

Il processo genitore lancia il processo figlio, il quale copia il valore di `value` nella sua area di memoria che non è condivisa con il processo padre. All'inizio tale valore è 0. A sua volta il processo figlio è costituito da un thread principale e da un thread figlio che esegue `runner()` che mette a 5 il valore di `value`. Poiché la memoria è condivisa tra il thread genitore e quello figlio, il valore di `value` è 5 anche per il thread genitore che è il processo figlio del processo genitore lanciato con il `main()`. Perciò alla riga C il processo figlio stampa 5, mentre alla riga P il processo genitore stampa 0, poiché il cambiamento di `value` è avvenuto nell'area di memoria del processo figlio (condivisa dal thread genitore e dal thread figlio), ma non è avvenuto nell'area di memoria del processo genitore che non è condivisa con il processo figlio.

In breve, il processo figlio ottenuto con una `fork` aggiorna la sua copia di `value`; quindi quando il controllo ritorna al processo genitore il valore di `value` del processo genitore rimane 0.

2. Siano dati 5 processi con durata dell'esecuzione e tempo d'arrivo espressi dalla seguente tabella:

Processo	Durata	Arrivo
P1	2	0
P2	6	1
P3	1	4
P4	4	7
P5	3	8

- (a) Si disegni lo schema GANTT (come nel libro di testo) che illustri l'esecuzione di questi processi con i seguenti algoritmi di schedulazione: FCFS (First Come First Serve), SRTF (Shortest-Remaining-Time-First, cioè Shortest-Job-First con prelazione), RR (Round-Robin) con quanto di tempo = 1.

Traccia di soluzione.

FCFS:

```
P1 P1 P2 P2 P2 P2 P2 P2 P3 P4 P4 P4 P4 P5 P5 P5
0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
```

SRTF:

```
P1 P1 P2 P2 P3 P2 P2 P2 P2 P5 P5 P5 P4 P4 P4 P4
0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
```

RR:

```
P1 P2 P1 P2 P3 P2 P2 P4 P5 P2 P4 P5 P2 P4 P5 P4
0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
```

- (b) Per i tre algoritmi precedenti si calcoli il tempo di completamento di ciascun processo.

Traccia di soluzione.

Il tempo di completamento e' la differenza tra il tempo di terminazione e il tempo d'arrivo, cioe' il tempo trascorso da quando il processo arriva nella coda a quando termina. Nella letteratura in inglese e' designato anche come TRT (turnaround time), cioe' tempo trascorso.

Per FCFS i tempi di completamento sono:

P1 2,

P2 7,

P3 5,

P4 6,

P5 8

Per SRTF i tempi di completamento sono:

P1 2,

P2 8,

P3 1,

P4 9,

P5 4

Per RR i tempi di completamento sono:

P1 3,

P2 12,

P3 1,

P4 9,

P5 7

- (c) Per i tre algoritmi precedenti si calcoli il tempo di attesa di ciascun processo.

Traccia di soluzione.

Il tempo di attesa e' la differenza tra il tempo di completamento e la durata, cioe' e' il tempo trascorso in attesa nella coda senza eseguire.

Per FCFS i tempi di attesa sono:

P1 0,

P2 1,

P3 4,

P4 2,

P5 5

Per SRTF i tempi di attesa sono:

P1 0,

P2 2,

P3 0,

P4 5,

P5 1

Per RR i tempi di attesa sono:

P1 1,

P2 6,

P3 0,

P4 5,

P5 4

- (d) Per i tre algoritmi precedenti si calcoli il tempo di attesa medio di tutti i processi.

Traccia di soluzione.

Tempo di attesa medio per FCFS: $(0+1+4+2+5)/5 = 2,4$.

Tempo di attesa medio per SRTF: $(0+2+0+5+1)/5 = 1,6$.

Tempo di attesa medio per RR: $(1+6+0+5+4)/5 = 3,2$.

Si noti che SRTF ottiene il tempo di attesa medio minimo come dalla teoria.

3. Si consideri uno schema d'indirizzamento a due livelli basato su pagine per indirizzi di memoria virtuale a 24 cifre binarie, con il seguente formato:

cifre 23-16	cifre 15-8	cifre 7-0
# pagina virtuale	# pagina virtuale	spiazzamento

Gl'indirizzi virtuali sono tradotti in indirizzi fisici di 16 cifre binarie, con il seguente formato:

cifre 15-8	cifre 7-0
# pagina fisica	spiazzamento

Gli elementi delle tavole delle pagine hanno 16 cifre binarie, con il seguente formato:

cifre 15-8	cifre 7 6 5 4 3 2 1 0
# pagina fisica	informazione varia

dove l'informazione varia e' definita come segue:

posizione 7 riservato sistema operativo

posizione 6 irraggiungibile

posizione 5 0

posizione 4 0

posizione 3 modificato in scrittura

posizione 2 usato

posizione 1 accessibile in scrittura

posizione 0 valido

Una traduzione da virtuale a fisico puo' fallire per uno di due motivi: pagina invalida (la pagina non e' presente in memoria fisica), oppure violazione d'accesso (la pagina e' presente in memoria fisica, ma l'accesso e' illegale).

- (a) Si calcoli la dimensione di una pagina.

Traccia di soluzione

Sia ETP l'acronimo per un elemento della tavola delle pagine (in inglese PTE)

La dimensione di una pagina è $2^8 \times \text{dimensione}(ETP) = 256 \times 2 = 512$ bytes, poiché lo spiazzamento è di 8 cifre binarie, e inoltre un elemento della tavola delle pagine (ETP) ha 16 cifre binarie, cioè 2 bytes.

- (b) Si calcoli la dimensione massima delle tavole delle pagine, tenendo conto di entrambi i livelli d'indirizzamento.

Traccia di soluzione

La dimensione massima della tavola delle pagine si presenta quando c'è una traduzione da virtuale a fisico per tutte le pagine nello spazio degli indirizzi. Si dovrebbero riempire tutti gli elementi in entrambi i livelli della tavola delle pagine. In questa tavola delle pagine a due livelli, ci sarà un pezzo della tavola delle pagine al primo livello, e 2^8 pezzi della tavola delle pagine al secondo livello per un totale di $1 + 2^8$ pezzi. Ogni pezzo contiene $2^8 \times \text{dimensione}(ETP) = 256 \times 2 = 512$ bytes (sono 2^8 perché gli indirizzi virtuali sia per il primo che per il secondo livello sono di 8 cifre binarie; inoltre un elemento della tavola delle pagine (ETP) ha 16 cifre binarie, cioè 2 bytes). Perciò la dimensione totale della tavola delle pagine completa è $(1 + 2^8) \times 512 = 131584$ bytes.

- (c) Spiegate testualmente e illustrate con un disegno lo schema proposto di risoluzione degli indirizzi da virtuale a fisico.

- (d) Si consideri il contenuto della memoria fisica nello schema allegato. Si assuma che il puntatore alla tavola di primo livello (Table Base Pointer) punti all'indirizzo $0x002000$.

Si determinino i risultati delle seguenti istruzioni di lettura (Load) e scrittura (Store) eseguite in modalita' utente.

- i. Load[0x0700FE],
- ii. Store[0x0700FE].

Gli indirizzi indicati sono virtuali e devono essere tradotti in indirizzi fisici.

Il risultato di una lettura e' un dato di 8 cifre binarie, se la lettura e' andata a buon fine, o una segnalazione d'errore. Il risultato di una scrittura e' un SI, se la lettura e' andata a buon fine, o una segnalazione d'errore.

Nel caso di errore se ne specifichi il tipo: pagina invalida oppure violazione d'accesso (per una scrittura senza il privilegio per scrivere, o per una lettura/scrittura da pagina riservata al sistema operativo).

Si mostrino in dettagli tutti i passi del meccanismo di traduzione degli indirizzi per arrivare alla risposta.

Traccia di soluzione.

Load[0x0700FE] produce *EE*.

Store[0x020731] produce violazione d'accesso per mancanza del privilegio di scrittura.

4. Si progetti un circuito sequenziale che soddisfa la seguente specifica:

- C'è un segnale binario d'ingresso X e un segnale binario d'uscita Z .
- Se la successione d'ingressi letti finora è la rappresentazione seriale in binario a partire dalla cifra più significativa di un numero intero positivo congruente a 0 modulo 5 (cioè divisibile per 5) e se essa inizia per 1, l'uscita Z vale 1, altrimenti l'uscita Z vale 0.

- (a) Si disegni il grafo delle transizioni di una macchina a stati finiti di tipo Mealy che corrisponde alla specifica. S'indichi lo stato iniziale.

Traccia di soluzione.

Si definiscano degli stati si che ricordino che la sottostringa letta sino a quel momento rappresenta un numero intero che diviso per 5 da' come resto il numero intero i , ad es. la macchina e' nello stato $s3$ quando la sottostringa letta sinora rappresenta in binario un numero la cui divisione per 5 produce come resto 3. Per definire le transizioni dallo stato si si rifletta a come cambia il valore di un numero binario estendendolo leggendo un altro 0 o un altro 1, e a come cambia il resto della divisione (modulo 5) di questo numero esteso.

S'introduca anche uno stato iniziale sa che sotto l'ingresso 0 va nello stato sn dove poi rimane per ogni ingresso (stringa malformata in base alla specifica).

Tavola delle transizioni e uscite della macchina a stati finiti:

I	SP	SF	U	
0	sa	sn	0	# sa stato iniziale
1	sa	s1	0	
-	sn	sn	0	
0	s1	s2	0	
1	s1	s3	0	
0	s2	s4	0	
1	s2	s0	1	
0	s3	s1	0	
1	s3	s2	0	
0	s4	s3	0	
1	s4	s4	0	
0	s0	s0	1	
1	s0	s1	0	

(b) Si minimizzi il numero degli stati della macchina di Mealy proposta.

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

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

- (e) 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.