

# Elementi di Architettura e Sistemi Operativi

Bioinformatica - Tiziano Villa

29 Settembre 2016

Nome e Cognome:

Matricola:

Posta elettronica:

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

1. Si consideri il seguente frammento d'implementazione d'una applicazione bancaria che realizza la sincronizzazione tra processi di accesso ai terminali bancari.

Si dica se funziona sempre, non funziona mai, o talvolta funziona talvolta no. Nel secondo e terzo caso, si spieghi il perché spiegando gli errori, e poi si mostri come possono essere corretti.

```
BankServer() {
    while (TRUE) {
        RiceviRichiesta(&op, &contoId1, &contoId2, &importo);
        if (op == trasferimento)
            ThreadFork(Trasferisci(contoId1, contoId2, importo));
        else if ...
    }
}
```

```
Trasferisci(contoId1, contoId2, importo) {
    conto[contoId1]->Lock();
    conto1 = LeggiConto(contoId1); /* accesso a disco */
    conto[contoId2]->Lock();
    conto2 = LeggiConto(contoId2); /* accesso a disco */
    if (conto1->saldo < importo) return ERROR;
    conto1->saldo -= importo;
    conto2->saldo += importo;
    ScriviConto(conto1); /* accesso a disco */
    ScriviConto(conto2); /* accesso a disco */
    conto[contoId1]->unlock();
    conto[contoId2]->unlock();
    return OK;
}
```

Traccia di soluzione.

Talvolta funziona, talvolta no. Ci sono due problemi:

- Il sistema puo' bloccarsi se ci sono dei trasferimenti simultanei circolari del tipo  $A \rightarrow B$  e  $B \rightarrow A$ ; ad es. un utente accede al conto A e blocca l'accesso ad altri prendendo il lucchetto di A, un altro utente accede al conto B e blocca l'accesso ad altri prendendo il lucchetto di B, per cui il primo utente non riesce ad accedere al conto B e il secondo utente non riesce ad accedere al conto A, ma ciascuno tiene il lucchetto del conto a cui ha avuto accesso senza rilasciarlo.
- Il sistema puo' bloccarsi anche quando una chiamata termina con ERROR, perche' non sono rilasciati i lucchetti.

Il codice seguente incorpora le soluzioni ai due problemi:

- Si acquisiscono i lucchetti nell'ordine dei loro numeri di conto, in modo che se si devono fare trasferimenti circolari e il numero di conto di A e' maggiore di quello di B, il primo utente (che avra' A e B come argomenti) supponiamo accedera' al conto A e poi al conto B, mentre il secondo utente (che avra' B e A come argomenti) cerchera' di accedere prima al conto A ma lo trovera' bloccato per cui aspettera' che il primo utente finisca e solo dopo accedera' ai conti A e B. In altri termini, si definisce un ordine di accesso ai conti che deve essere rispettato da tutti, in modo che non ci siano accessi incrociati che si bloccano a vicenda.
- Prima di terminare una chiamata con ERROR si rilasciano i lucchetti.

La soluzione di mettere un lucchetto di guardia all'acquisizione di entrambi i lucchetti e' inefficiente perche' serializza l'inizio delle procedure di trasferimento. La soluzione di mettere un lucchetto globale a tutta la procedura di trasferimento e' inaccettabilmente inefficiente. Naturalmente il codice trascura molte piccole ottimizzazioni (come verificare che i numeri di conto non coincidano etc.).

```

Trasferisci(contoId1, contoId2, importo) {
    if (contoId1 > contoId2) {
        conto[contoId1]->Lock();
        conto[contoId2]->Lock();
    } else {
        conto[contoId2]->Lock();
        conto[contoId1]->Lock();
    }
    conto1 = LeggiConto(contoId1); /* accesso a disco */
    conto2 = LeggiConto(contoId2); /* accesso a disco */
    if (conto1->saldo < importo) {
        conto[contoId1]->unlock();
        conto[contoId2]->unlock();
        return ERROR;
    }
    conto1->saldo -= importo;
    conto2->saldo += importo;
    ScriviConto(conto1); /* accesso a disco */
    ScriviConto(conto2); /* accesso a disco */
    conto[contoId1]->unlock();
    conto[contoId2]->unlock();
    return OK;
}

```

2. Si consideri una memoria con una cache. Il tempo di accesso della cache  $T_c$  e' di 100 ns e il tempo di accesso della memoria  $T_m$  di 1200 ns. Se il tempo di accesso effettivo  $T_e$  del sistema memoria+cache e' del 10% maggiore del tempo di accesso della cache, quale deve essere la percentuale di successo  $S$  ?

Traccia di soluzione.

La formula per il tempo di accesso effettivo e'

$$T_e = S \times T_c + (1 - S)(T_m + T_c),$$

dove  $T_c = 100 \text{ ns}$ ,  $T_e = 1,1 \times T_c$ , and  $T_m = 1200 \text{ ns}$ .

Sostituendo i valori si ha

$$1,1 \times 100 = 100S + (1 - S)(1200 + 100)$$

$$110 = 100S + 1300 - 1300S$$

$$1200S = 1190$$

$$S = 119/120 \approx 99,1\%.$$

3. A volte serve un'istruzione che non fa nulla, chiamata NOP (in inglese, NO OPERATION). Questa istruzione e' prelevata, decodificata ed eseguita, ma la fase di esecuzione consiste nel non fare nulla.

Quale delle seguenti tre istruzioni in binario nel linguaggio macchina LC-3 potrebbe essere usata correttamente come NOP ? Si argomenta la risposta per tutte e tre le istruzioni.

- (a) 0001 001 001 1 00000
- (b) 0000 111 000000001
- (c) 0000 000 000000000

Traccia di soluzione.

- (a) Add R1, R1, #0

Non e' una NOP perche' assegna i codici di condizione (CC) e quindi cambia lo stato del processore.

- (b) BRnzp #1

Non e' una NOP, perche' salta incondizionatamente all'indirizzo successivo a quello nel contatore di programma, e quindi cambia il flusso logico del programma.

- (c) E' una NOP, perche' e' un salto che non e' mai eseguito.

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

- Ci sono due variabili binarie in ingresso  $x$  e  $y$ , e due variabili binarie in uscita  $e$  e  $z$ .
- L'uscita  $e = 0$  al ciclo  $\tau_i$ , se al ciclo  $\tau_{i-1}$  la somma dei valori di  $x$  e  $y$  era 0, altrimenti  $e = 1$  al ciclo  $\tau_i$ .
- L'uscita  $z = 0$  al ciclo  $\tau_i$ , se la somma di tutti i valori di  $x$  e  $y$  dall'inizio  $\tau_0$  al ciclo precedente  $\tau_{i-1}$  incluso era 0, altrimenti  $z = 1$  al ciclo  $\tau_i$ .

(a) Si disegni il grafo delle transizioni di una macchina a stati finiti che realizza la specifica. S'indichi lo stato iniziale.

Data la specifica, la soluzione sara' una macchina di Moore o di Mealy ?

Traccia di soluzione.

La specifica richiede una macchina di Moore. Segue la tavola delle transizioni.

I	SP	SF	U
00	s00	s00	00
01	s00	s11	00
10	s00	s11	00
11	s00	s11	00
00	s11	s01	11
01	s11	s11	11
10	s11	s11	11
11	s11	s11	11
00	s01	s01	01
01	s01	s11	01
10	s01	s11	01
11	s01	s11	01

- (b) Si minimizzi il numero degli stati della macchina proposta, applicando l'algoritmo di minimizzazione degli stati.



- (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. Si esegua e mostri la minimizzazione con le mappe di Karnaugh.

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