

# Elementi di Architettura e Sistemi Operativi

Bioinformatica - Tiziano Villa

5 Settembre 2011

Nome e Cognome:

Matricola:

Posta elettronica:

problema	punti massimi	i tuoi punti
problema 1	8	
problema 2	6	
problema 3	8	
problema 4	8	
totale	30	

1. Rispondere in modo preciso ma conciso alle seguenti domande.

(a) Si descrivano le quattro condizioni necessarie perche' si abbia uno stallo.

Traccia di soluzione.

- i. Mutua esclusione (almeno una risorsa non condivisibile).
- ii. Possesso e attesa (un processo in possesso di almeno una risorsa e' in attesa di almeno un'altra risorsa detenuta da altri).
- iii. Impossibilita' di prelazione (una risorsa puo' essere rilasciata dal processo che la detiene solo volontariamente, non si puo' sottrargliela).
- iv. Attesa circolare (c'e' una catena chiusa di processi che attendono risorse detenute da altri processi).

(b) Si definisca il grafo di assegnazione delle risorse e se ne motivi l'uso.

Traccia di soluzione.

Si veda la Sez. 7.2.2 del Silberschatz.

Quale relazione esiste tra l'esistenza di cicli nel grafo delle risorse e la possibilità di uno stallo ?

Traccia di soluzione.

L'assenza di cicli nel grafo di assegnazione delle risorse implica l'assenza di situazioni di stallo. Ma la presenza di cicli non implica necessariamente uno stallo.

- (c) Per ciascuna delle quattro condizioni necessarie per uno stallo si discutano i modi di prevenirla.

Traccia di soluzione.

- i. Le risorse condivisibili non richiedono la mutua esclusione, ma non si può eliminare la mutua esclusione sulle risorse non condivisibili.
  - ii. Si può usare un protocollo
    - A. in cui ogni processo prima d'iniziare la propria esecuzione ottiene tutte le risorse che gli servano; oppure
    - B. in cui un processo richiede risorse solo se non ne possiede (se ne possiede, prima di richiederne altre deve rilasciare quelle che possiede).
- Si veda la Sez. 7.4.2.
- iii. S'introduce una qualche forma di prelazione nei casi in cui lo stato si può salvare e recuperare facilmente. Si veda la Sez. 7.4.3.
  - iv. Per impedire l'attesa circolare si può imporre un ordine totale all'insieme di tutti i tipi di risorse e un ordine crescente di numerazione alle risorse richieste da ciascun processo. Un protocollo è il seguente: un processo può richiedere inizialmente qualsiasi quantità di una certa risorsa, ad es. della risorsa  $R_i$ , dopo di che il processo può richiedere quantità della risorsa  $R_j$  solo se  $R_j > R_i$  nell'ordine totale dato.

- (d) Si definisca quando uno stato si dice sicuro. Qual e' la relazione tra uno stato sicuro, uno insicuro e uno di stallo ?

Traccia di soluzione.

Uno stato si dice sicuro se c'e' un ordine in base al quale il sistema e' in grado di assegnare le risorse richieste da ciascun processo (eventualmente assegnandogli le risorse usate dai processi che vengono prima nell'ordine e che quindi si puo' supporre che abbiano finito).

Se non esiste tale ordine lo stato del sistema si dice insicuro. Uno stato di stallo e' uno stato insicuro, ma non tutti gli stati insicuri sono di stallo. Uno stato insicuro potrebbe condurre a uno stallo, perche' il sistema operativo non puo' impedire ai processi di richiedere risorse in modo da causare uno stallo.

Si veda la Fig. 7.5 nella Sez. 7.5.1.

- (e) Se un sistema e' in uno stato insicuro, tale sistema deve finire inevitabilmente in uno stato di stallo ? In altri termini, e' possibile che i processi del sistema completino la loro esecuzione senza che il sistema entri in uno stato di stallo ?

Traccia di soluzione.

Uno stato insicuro non conduce necessariamente a uno stallo, ma d'altro canto non possiamo garantire che non si arrivi a uno stallo. Quindi e' possibile che in un sistema in uno stato insicuro tutti i processi completino senza uno stallo.

(f) Si consideri un sistema con 12 risorse e 3 processi  $P_0, P_1, P_2$  con la seguente distribuzione di risorse:

$P_0$  ha bisogno al massimo di 10 risorse ed ora detiene 5 risorse;

$P_1$  ha bisogno al massimo di 4 risorse ed ora detiene 2 risorse;

$P_2$  ha bisogno al massimo di 9 risorse ed ora detiene 3 risorse.

In che stato si trova il sistema ? E' inevitabile che il sistema finisca in stallo ? E' possibile che il sistema non finisca in stallo ?

Traccia di soluzione.

In questo momento nel sistema ci sono 2 risorse libere. Il sistema e' in un stato insicuro perche' pur se  $P_1$  potrebbe acquisire le due risorse libere e poi terminare, liberando 4 risorse, non potremmo ancora garantire che i processi  $P_0$  e  $P_2$  terminino. Tuttavia sarebbe possibile che un processo rilasciasse risorse prima di richiederne altre. Ad esempio, il processo  $P_2$  potrebbe rilasciare una risorsa, portando a 5 il numero delle risorse libere. Questo permetterebbe al processo  $P_0$  di completare, portando a 10 il numero delle risorse libere, permettendo infine al processo  $P_2$  di completare.

2. Si consideri il seguente scenario di lavanderia automatica. Un cliente in arrivo depone delle monete in una di due postazioni di servizio e richiede una macchina per lavare. Le due postazioni sono collegate a un calcolatore centrale che assegna automaticamente una macchina disponibile e produce un cartellino con il numero della macchina assegnata. Il cliente utilizza la macchina indicata, che alla fine del ciclo di lavaggio informa il calcolatore centrale di essere di nuovo disponibile. Il calcolatore centrale gestisce un vettore `disponibile[NMACCHINE]`, dove la posizione  $i$  e'  $\neq 0$  se la macchina  $i$ -esima e' disponibile. La costante `NMACCHINE` indica il numero di macchine totali nella lavanderia. C'e' un semaforo `nlibere` che indica quante macchine sono disponibili. Il codice per assegnare e rilasciare le macchine segue. Il vettore `disponibile` e' inizializzato con tutti uno, la variabile `nlibere` e' inizializzata a `NMACCHINE`.

```
int allocazione() {
    int i;
    P(nlibere);
    for (i = 0; i < NMACCHINE; i++) {
        if (disponibile[i] != 0) {
            disponibile[i] = 0;
            return i;
        }
    }
}

rilascio(int macchina) {
    disponibile[macchina] = 1;
    V(nlibere);
}
```



(a) Si analizzi il codice precedente spiegandone il funzionamento.

- (b) E' riportato che occasionalmente, quando due persone richiedono una macchina nello stesso momento, si ritrovano assegnata la medesima lavatrice. Vi si chiede di analizzare il codice e spiegare quando cio' possa succedere.

Traccia di soluzione.

Il problema insorge perche' la manipolazione del vettore `disponibile` non e' protetta da un meccanismo di mutua esclusione. Puo' succedere che il processo di un cliente *A* esegua fino a prima dell'assegnamento `disponibile[i] = 0` e poi sia interrotto; e che poi sia attivato il processo di un cliente *B* che riesca ad eseguire tutta la procedura di allocazione ricevendo la macchina *i*; infine il processo del cliente *A* riprende e completa l'esecuzione ricevendo anch'esso la macchina *i*.

In altri termini, il semaforo `nlibere` garantisce che ci siano abbastanza macchine disponibili per ogni processo che raggiunge l'assegnamento `disponibile[i] = 0`, ma non garantisce che un processo abbia accesso esclusivo alle variabili di stato.

- (c) Si modifichi il codice precedente aggiungendo due istruzioni per eliminare l'inconveniente riportato.

Traccia di soluzione.

La soluzione e' di aggiungere un semaforo di mutua esclusione `Mutex` inizializzato a 1.

```
int allocazione() {
    int i;
    P(nlibere);
    P(Mutex);
    for (i = 0; i < NMACCHINE; i++) {
        if (disponibile[i] != 0) {
            disponibile[i] = 0;
            V(Mutex);
            return i;
        }
    }
}

rilascio(int macchina) {
```

```
P (Mutex) ;  
disponibile[macchina] = 1;  
V (Mutex) ;  
V (nlibere) ;  
}
```

Si noti che non e' necessario introdurre la mutua esclusione nella procedura `rilascio`, ma e' una buona pratica proteggere con la mutua esclusione tutte le manipolazioni delle variabili di stato.

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) Spiegate testualmente e illustrate con un disegno lo schema proposto di risoluzione degli indirizzi da virtuale a fisico.
- (b) Spiegate il meccanismo di "copia su scrittura" ("copy on write").

Traccia di soluzione.

Si veda la Sez. 9.3 del libro di testo.

L'idea è che processi genitori e figli inizialmente condividono le pagine. Se un processo genitore o figlio scrive su una pagina condivisa il sistema crea una copia di tale pagina.

- (c) 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 calcolino i valori prodotti dalle seguenti istruzioni di lettura (Load) e scrittura (Store) eseguite in modalita' utente.

- i. Load[0x0C2345],
- ii. Store[0x000115].

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

Il valore prodotto da una lettura e' un dato di 8 cifre binarie, se la lettura e' andata a buon fine, o una segnalazione d'errore. Il valore prodotto da 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[0x0C2345] produce segnalazione d'errore: pagina invalida.

Store[0x000115] produce SI.

Nota: ogni elemento della tavola occupa 2 bytes di memoria, percio' l'indice  $i$  punta alle posizioni di memoria  $2i$  e  $2i+1$  a partire dall'indirizzo di base (nel primo passo dato dall'inizio della tavola); si veda la figura allegata. Ad es. nel caso di Load[0x0C2345] a partire dall'indirizzo  $0x2000$  (dato dal registro di base) l'indice 0C (C esadecimale sta per 12 decimale) richiede che si scorra nella memoria di  $2*12=24$  e  $2*12+1=25$  posizioni fino agli indirizzi  $0x2018$  (da  $18_{16} = 24_{10}$ , si ha  $2000_{16} + 18_{16}$ ) e  $0x2019$  (da  $19_{16} = 25_{10}$ , si ha  $2000_{16} + 19_{16}$ ) che contengono entrambi 00. E cosi' per il secondo livello d'indirizzamento.

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 contiene un numero di 1 che è multiplo di 5, 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.

E' un contatore con 5 stati, ciascuno dei quali va nel successivo con ingresso/uscita 1/0, tranne che dal quarto al quinto si va con 1/1. Inoltre ogni stato ha un autoanello con ingresso/uscita 0/0.

Piu' precisamente le seguenti due soluzioni sono accettabili. La prima soluzione e' la piu' ragionevole; nella seconda anche il caso di zero 1 e' considerato un multiplo di 1 (con molteplicita' 0).

0	st0	st0	0
1	st0	st1	0
0	st1	st1	0
1	st1	st2	0
0	st2	st2	0
1	st2	st3	0
0	st3	st3	0
1	st3	st4	0
0	st4	st4	0
1	st4	st5	1
0	st5	st5	1
1	st5	st1	0

**oppure**

0	st0	st0	1
1	st0	st1	0
0	st1	st1	0
1	st1	st2	0
0	st2	st2	0
1	st2	st3	0
0	st3	st3	0
1	st3	st4	0
0	st4	st4	0
1	st4	st0	1



(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.