SISTEMI OPERATIVI 2020/2021

SCRITTO N 3 – 10 GIUGNO 2021

(Docente: Andrea Lanzi)

1. **In un sistema operativo UNIX, le chiamate di sistema della “famiglia” exec() ( execl(), execv(), etc, …):**

**(2 punti)**

* 1. Sono gli unici meccanismi di creazione di nuovi processi
  2. Sono i principali meccanismi di creazione di nuovi processi
  3. Causano la terminazione del processo in corso e l’avvio di un nuovo processo
  4. Causano la sostituzione di tutte le informazioni del processo in corso con un processo nuovo
  5. Riportano come risultato il PID del nuovo processo
  6. Riportano come risultato il valore restituito dalla funzione main() dell’eseguibile modificato
  7. Nessuna delle affermazioni precedenti è corretta

Domanda 1 - Corretta (g)

1. **Chiamata di sistema Fork (3 punti)**

In un sistema operativo UNIX, la chiamata di sistema fork():

1. Sostituisce al processo esistente un nuovo processo, identico in tutto e per tutto al processo chiamante, tranne che nel Process Control Block (PCB)
2. Sostituisce al processo esistente un nuovo processo, costruito sulla base del file eseguibile passato come argomento della chiamata, mantenendo il PCV del processo originale
3. Divide il processo esistente in due processi che condividono il PCB: il primo identico al processo originale, il secondo costruito a partire dall’eseguibile passato come argomento alla chiamata
4. Genera un novo processo, identico in tutto e per tutto al processo chiamante, tranne che nel Process Control Block (PCB)
5. Genera un nuovo processo, lanciando una nuova istanza del processo chiamante a partire dal file eseguibile corrispondente
6. È l’unico meccanismo disponibile con cui un processo può generare un altro processo
7. È il meccanismo più frequentemente usato dai processi per generare nuovi processi.

Domanda 2 - Corrette (d, f)

1. **Scheduler Round Robin (3 punti)**

In uno scheduler Round Robin:

* 1. l’avanzamento relativo dei processi è sincronizzato e dipende solo dai tempi di arrivo, dato che tutti i processi ricevono ciclicamente a turno un quanto di tempo, nell’ordine in cui sono entrati nel sistema
  2. l’avanzamento relativo dei processi è di fatto asincrono e imprevedibile perché i processi possono fare richieste bloccanti di dati che arrivano asincronamente dall’esterno del sistema
  3. l’avanzamento relativo dei processi è di fatto asincrono e imprevedibile perché influenzato dalle interruzioni che il sistema riceve
  4. ogni processo ha un quanto di tempo di durata differente, che deve essere nota in anticipo
  5. ogni processo ha un quanto di tempo di durata differente, determinata con una media sulla base delle esecuzioni precedenti
  6. se la durata del quanto di tempo tende a infinito, il comportamento tende a quello di uno scheduler FCFS (First Come First Served)
  7. se la durata del quanto di tempo tende a infinito, il comportamento tende a quello di uno scheduler SPN (Shortest Process Next)

Domanda 3 - Corrette(b, c, f)

1. **Scheduling e starvation**

**(2 punti)**

In un sistema di scheduling, quali dei seguenti algoritmi possono causare starvation (attesa indefinita) ?

1. First Come First Served (FIFO)
2. Last Come First Server (LIFO)
3. Round Robin
4. Round Robin con quanto di tempo di durata illimitata
5. Shortest Process Next (SPN)
6. L’algoritmo utilizzato dallo scheduler reale di Linux

Domanda 4 – Corrette (b, e)

1. **Gestione della memoria con segmentazione (3 punti)**

La gestione della memoria con segmentazione

* 1. è una generalizzazione della paginazione
  2. è una generalizzazione del partizionamento statico
  3. è una generalizzazione del partizionamento dinamico
  4. è soggetta a frammentazione esterna
  5. è soggetta a frammentazione interna
  6. non è soggetta ad alcuna frammentazione
  7. peggiora l’organizzazione interna dei processi, forzando la separazione di blocchi di codice e di dati
  8. razionalizza l’organizzazione interna dei processi, consentendo la separazione di blocchi e di codice dati
  9. aumenta il consumo di memoria, forzando in ogni processo la suddivisione di blocchi di codice e di dati in segmenti differenti
  10. aumenta il consumo di memoria, forzando più precessi a condividere blocchi di codice e di dati
  11. riduce il consumo di memoria, consentendo a più processi di condividere blocchi di codice e di dati
  12. riduce il consumo di memoria, consentendo a più processi di condividere blocchi di codice e di dati
  13. non è descritta correttamente da nessuna delle affermazioni precedenti

Domanda 5 – Corretta (c,d,h,k)

1. **Traduzione di indirizzi di memoria con paginazione (5 punti)**

Considerare un sistema di traduzione da indirizzamento logico ad indirizzamento fisico realizzato mediate paginazione. Lo spazio logico di un programma è costituito da un massimo di 64 byte, suddivise in pagine da 4 byte. La memoria fisica è costituita da 256byte.

2^6 = 64; 2^2 = 4; 2^8 = 256;

* 1. Da quanti bit sono costituiti gli indirizzi logici e gli indirizzi fisici?

Logici = 4 bit + 2bit di offset; Fisici = 6 bit + 2 bit di offset;

* 1. Da quanti bit sono costituiti i numeri di pagina?

Pagine = 2 bit

* 1. Da quanti bit sono costituiti i numeri di frame?

Frame = 256/4 = 64; 2^6 = 64; quindi frames = 6 bits

* 1. Ad un dato istante la tabella delle pagine (page table) di un processo è la seguente

Num pagina logica Num pagina fisica

1. 12
2. 1
3. 17
4. 62
5. 11
6. 16
7. 61
8. 12

Tradurre in indirizzi fisici i seguenti indirizzi logici: 0, 2, 4, 9, 19, 11, 22, 32, 30, 26, 23, 36

1. **Traduzione indirizzi di memoria con segmentazione**

**(5 punti)**

Considerare un sistema con memoria virtuale e segmentazione con parola da un byte.

Supponendo di avere indirizzi logici di 10 bit, i cui primi 4 più significativi indicano il numero di segmento, dire:

1. Qual è la massima grandezza possibile per un segmento

2^6 = 64 dato che 10 bit meno 4 uguale a 6. Quindi 2 elevato alla 6 = 64.

1. Di quanti segmenti al più può essere composto un processo

2^4 = 16 dato che i msb sono formati da 4 bit. Quindi 16.

Supponendo che il processo attualmente in esecuzione sia comporto da 5 segmento S0, S1, S2, S3, S4 e che ad un dato istante la sua tabella dei segmenti sia la seguente:

Num di segmento Lunghezza Base

0 10 200  
1 20 100  
2 6 252  
3 32 720  
4 32 683

Tradurre in indirizzi fisici i seguenti indirizzi logici: 9, 132, 79, 64, 259, 135, 320

9 = seg 0; 200+9 = 209;

132 = seg 2; 252 + 4(bit risultanti) = 256;

1. **DMA (2 punti)**

In un elaboratore, il DMA (Direct Memory Access):

* 1. È una tecnica che demanda l’I/O ad un apposito modulo software
  2. È una tecnica che demanda l’I/O ad un apposito modulo hardware
  3. È una tecnica che evita il ricorso a qualsiasi interrupt
  4. È una tecnica che utilizza un numero minimo di interrupt
  5. È l’attributo dei segmenti di memoria che possono essere acceduti da ogni processo
  6. È lo stato a cui passa un processo uscendo dallo stato Suspend

Domanda 8 – Corrette (b, d)

1. **Implementazione di un semaforo binario e di un semaforo generico (6 punti)**

Un particolare sistema di supporto alla mutua esclusione utilizza delle variabili di tipo lock t e tre chiamate

**init\_lock (lock\_t \*l)** inizializza la variabile di lock allo stato libero, e deve essere obbligatoriamente chiamata prima di poter utilizzare la variabile in questione

**get\_lock (lock\_t \*l)** consente al processo chiamante di acquisite il lock e proseguire, se il lock era libero, altrimenti blocca il processo su una coda FIFO associata alla variabile di lock.

**release\_lock(lock\_t \*l)**  sblocca il primo processo in coda, se presente, altrimenti riporta il lock allo stato libero

Utilizzando il sistema descritto implementare:

* + - Un sistema di semaforo generico, scrivendo il codice per la struttura dati sem\_t e le routine init(sem\_t \*l, int n), wait(sem\_t \*l) e signal(sem\_t \*l)
    - Un sistema di semaforo binario, scrivando il codice per la struttura dati bsem\_t e le routine binit(bsem\_t \*l, bool b), bwait(sem\_t \*l) e bsignal(sem\_t \*l)