Samuele Dell'oca

+20/1/28+

Sistemi Operativi TP 2020
Prova Scritta

Nome e Cognome: SAMUELE DELL'OCA

Tempo a disposizione: 90 minuti / Documentazione ammessa: 1 foglio A5 manoscritto.

(punti totali: 77.5/89)

Domande a risposta multipla [75 punti]

Ogni domanda può avere una o più risposte corrette. Una domanda completamente corretta è valutata 3 punti.

Domanda 1 ◇ I segnali POSIX...

- [ ] sono sincroni
- [x] possono essere gestiti in modo sincrono
- [ ] vengono utilizzati per le chiamate di sistema
- [x] sono asincroni

3/3

Domanda 2 ◇ L'algoritmo di scheduling sui sistemi Unix tradizionali...

- [ ] esegue i processi con ordine FCFS
- [x] esegue i processi con ordine Round-Robin
- [x] implementa una schedulazione con priorità dinamiche
- [ ] implementa una schedulazione con priorità statiche

1.5/3

Domanda 3 ◇ In un sistema che implementa lo scheduling Fair Share ho 3 utenti X, Y e Z e un programma per il calcolo della meteo P con tempo di esecuzione costante T. Contemporaneamente, l'utente X mette in esecuzione una istanza del programma P, Y mette in esecuzione due istanze del programma P, mentre Z mette in esecuzione una istanza del programma P. In una situazione del genere...

X 1
Y 2
Z 1

- [ ] tutti gli utenti avranno ricevuto complessivamente lo stesso tempo CPU
- [x] a fine esecuzione ogni istanza avrà ricevuto complessivamente lo stesso tempo CPU
- [x] a fine esecuzione l'utente Y avrà ricevuto complessivamente più CPU dell'utente X
- [ ] tutte le istanze del programma P di tutti gli utenti termineranno allo stesso momento

3/3

Domanda 4 ◇ I processi di tipo *CPU-Bound*...

- [ ] non possono essere schedulati in un sistema con requisiti realtime
- [ ] vengono eseguiti con una priorità più alta rispetto agli altri processi
- [ ] effettuano numerose operazioni di input-output che bloccano la CPU
- [x] necessitano tipicamente di un algoritmo di scheduling pre-emptive

3/3

Domanda 5 ◇ Nei primi esempi presenti nel *workbook* abbiamo introdotto una funzione detta *cushion*. In questa funzione era dichiarato un array di *char* e poi veniva chiamata la funzione del *thread*. Quali affermazioni sono vere?

- [x] non devo ritornare alla funzione *cushion* al termine dell'esecuzione del thread
- [ ] l'array del cushion viene condiviso da tutti i thread
- [ ] durante l'esecuzione di un thread l'array viene sovrascritto con le istruzioni del thread
- [x] la dimensione minima dell'array dipende da quante chiamate ad altre funzioni faccio all'interno del thread che sta per essere eseguito

0/3

Domanda 6 ◇ L'algoritmo Shortest Remaining Time Next (SRTN)...

- [x] equivale a SJF se l'esecuzione dei processi non viene sospesa o prelazionata (pre-emption)
- [ ] favorisce l'equità (*fairness*)
- [x] può portare a una situazione di starvation
- [x] necessita di una stima del tempo totale di esecuzione dei processi

3/3

Domanda 7 ◇ Per implementare la *pre-emption*...

- [ ] devo minimizzare il tempo di risposta del sistema
- [ ] ogni thread deve fare periodicamente uno *yield*
- [ ] assegno una priorità più bassa ai processi che devono essere interrotti periodicamente
- [x] posso utilizzare degli *interrupt*

3/3

Domanda 8 ◇ Nella libreria *bthread*...

- [x] i thread appena creati si trovano nello stato $\_\_BTHREAD\_READY$
- [ ] i thread che si trovano nello stato $\_\_BTHREAD\_RUNNING$ non possono essere interrotti
- [ ] ogni thread ha la sua istanza dello scheduler
- [ ] i thread che sono in attesa di essere eseguiti sono nello stato $\_\_BTHREAD\_BLOCKED$

3/3

Domanda 9 ◇ In un sistema *realtime*...

- [ ] non posso utilizzare la pre-emption
- [ ] il risultato logico dei task è meno importante rispetto alle deadline
- [ ] i task si eseguono più rapidamente rispetto a un sistema non-realtime
- [x] i task si devono completare entro scadenze temporali precise

3/3

Domanda 10 ◇ Il gestore (*handler*) di un segnale POSIX...

- [ ] permette di gestire i segnali in modo sincrono
- [x] può essere collegato a più segnali diversi
- [ ] viene eseguito in modo sincrono
- [ ] è richiamato quando un segnale viene bloccato nella maschera

3/3

Domanda 11 ◇ Per implementare il multi-tasking preemptive è necessario disporre di un meccanismo che...

- [ ] garantisca che le istruzioni privilegiate vengano eseguite su un core diverso rispetto a quelle non privilegiate
- [x] garantisca che il kernel possa ottenere periodicamente il controllo della CPU

3/3

- [ ] garantisca che il kernel venga eseguito con una priorità più alta rispetto ai processi utente
- [ ] permetta ai processi di rilasciare volontariamente la CPU (yield)

Domanda 12 ◇ Una *policy* di scheduling basata sulla *hard-affinity*...

- [ ] permette allo scheduler di cambiare il core assegnato a un thread
- [x] obbliga lo scheduler ad assegnare un thread sempre allo stesso core

3/3

- [ ] è più efficace della *soft-affinity* per risolvere i problemi di bilanciamento del carico
- [ ] non si applica a processi/thread realtime

Domanda 13 ◇ Lo *slack-time*...

- [ ] di regola è zero nei sistemi che utilizzano gli algoritmi RMA/RMS o EDF
- [x] dipende dal tempo di esecuzione di un task e dalla sua deadline

3/3

- [ ] dipende dal tempo di esecuzione di un task e dalla sua periodicitâ
- [x] può essere uguale a zero

Domanda 14 ◇ La funzione *longjmp*...

- [x] modifica il contenuto dei registri
- [x] modifica il contenuto dello stack

0/2

- [ ] modifica la maschera dei segnali
- [ ] ritorna 0 solo alla prima esecuzione
- [x] *Nessuna risposta è giusta.*

Domanda 15 ◇ Un processo X si trova nello stato ready...

- [x] quando è pronto per essere eseguito
- [ ] quando attende una risorsa

3/3

- [x] quando una risorsa che stava aspettando diventa disponibile
- [ ] quando termina e il padre chiama **waitpid**

Domanda 16 ◇ In un sistema che implementa un'architettura di tipo NUMA...

- [ ] i tempi di accesso alla memoria non dipendono dalla presenza di cache
- [x] ogni CPU dispone di una parte privata di memoria

1.5/3

- [x] i tempi di accesso alla RAM dipendono dalla CPU che effettua la richiesta
- [ ] i tempi di accesso alla memoria sono costanti

Domanda 17 ◇ Il *throughput*...

- [ ] determina la velocità con cui lo scheduler riesce a fare *context-switch*
- [ ] deve essere prevedibile e regolare
- [ ] deve essere minimizzato sui sistemi *batch*
- [x] dipende dall'algoritmo di scheduling

3/3

Domanda 18 ◇ Subito dopo un **fork**, (quindi prima di un eventuale **exec**)...

- [x] il processo padre e il processo figlio eseguono lo stesso programma
- [ ] il processo figlio può accedere alle variabili del processo padre
- [ ] il processo padre può accedere alle variabili del processo figlio
- [ ] il PID del processo figlio è uguale a quello del padre

3/3

Domanda 19 ◇ Nello scheduling multi-core / multi-processore, il concetto di *locality*

- [ ] dipende dalla priorità di scheduling di un processo/thread
- [ ] dipende dalla quantità di memoria installata
- [ ] può essere temporale oppure dimensionale
- [x] dipende dagli accessi alla memoria fatti dal programma

3/3

Domanda 20 ◇ Il tempo medio di elaborazione...

- [x] dipende dal tempo di esecuzione
- [x] dipende dall'algoritmo di scheduling utilizzato
- [ ] è uguale al tempo di esecuzione se utilizzo l'algoritmo FCFS
- [ ] è identico nel caso di due task con lo stesso tempo di esecuzione

3/3

Domanda 21 ◇ Nella libreria *bthread* l'esecuzione dei thread avviene...

- [ ] in vero parallelismo
- [x] su un solo core/CPU
- [x] in pseudo-parallelismo
- [ ] in modalità pre-emptive utilizzando gli interrupt della CPU

1.5/3

Domanda 22 ◇ Un processo zombie...

- [x] non viene più schedulato dal sistema
- [ ] è un processo in esecuzione in background in una sessione separata
- [ ] è un programma malevolo simile a un virus
- [ ] è in esecuzione in background ma non è possibile terminarlo

3/3

Domanda 23 ◇ Uno scheduler *MLFQ*...

- [x] utilizza un sistema di priorità variabile (dinamico)
- [x] favorisce i processi di tipo *IO-Bound*

3/3

- [ ] suddivide il tempo CPU in modo equo tra gli utenti
- [ ] favorisce i processi di tipo *CPU-Bound*

Domanda 24 ◇ Un segnale realtime...

- [ ] ha priorità più alta rispetto ad un segnale non-realtime
- [ ] viene consegnato più rapidamente rispetto a un segnale *normale*

3/3

- [x] può essere bloccato modificando la maschera
- [x] può essere associato ad un gestore (handler)

Domanda 25 ◇ La *schedulabilità* di un sistema realtime...

- [x] dipende dal tempo di esecuzione di ogni task
- [x] dipende dal numero di CPU/core

3/3

- [ ] concerne solo i task periodici
- [ ] dipende solo dal numero di task

Vero o falso? [6 punti]

Domanda 26 Posso utilizzare la funzione *printf* in un gestore di segnale perché è definita come *async safe*.

0/1 ☒ Vero ☒ Falso

Domanda 27 Lo scheduling a lotteria non permette di implementare un sistema di priorità dinamiche.

1/1 ☐ Vero ☒ Falso

Domanda 28 Lo scheduling *SJF* non è adatto per programmi interattivi dove è richiesto l'input dell'utente.

1/1 ☒ Vero ☐ Falso

Domanda 29 Una coda vuota *TQueue* vuota è rappresentata da un oggetto *TQueueNode* che punta a sè stesso.

1/1 ☐ Vero ☒ Falso

Domanda 30 La funzione *setjmp* permette di salvare il contenuto dello stack all'interno di un buffer (*jmpbuf*).

0/1 ☒ Vero ☒ Falso

Domanda 31 La funzione *tqueue_pop* libera la memoria associata al puntatore *data*.

1/1 ☐ Vero ☒ Falso

Scheduling realtime [9 punti]

Domanda 32 Considera un sistema realtime con i seguenti 4 task periodici:

- A, tempo di esecuzione 1, periodicità 5
- B, tempo di esecuzione 2, periodicità 6
- C, tempo di esecuzione 1, periodicità 7
- D, tempo di esecuzione 1, periodicità 11

	A	B	C	D
×	1	2	1	1
P	5	6	7	11

$u = \frac{1}{5} + \frac{2}{6} + \frac{1}{7} + \frac{1}{11} \simeq 76,7\% \Rightarrow$ PUÒ ESSERE ESEGUITO SU UN CPU/CORE (< 100%)

$4(2^{\frac{1}{4}} - 1) \simeq 75,7\%$

Scegli l'algoritmo **più semplice/opportuno** tra i due visti durante il corso (RMA o EDF).
► **Indica chiaramente l'algoritmo scelto sulla riga tratteggiata sotto alla tabella e motiva la tua scelta su un foglio a parte.**
► **Completa la tabella indicando l'ordine di scheduling in base all'algoritmo scelto.** *Nota: considera che si tratta di un sistema non-preemptive con una sola CPU/un solo core.* **[3 punti]**

Tempo (ms) →	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
A	X					X					X					X					X					X				X			
B	X	X					X	X					X	X					X	X					X	X					X	X	
C	X							X							X							X							X				
D	X											X											X										
EDF	A	B	B	C	D	A	B	B	C		A	D	B	B	C	A			B	B	A	C	D		B	B	A		C		A	B	B

76,7% > 75,7% => DEVO USARE PER FORZA EDF ☐w ☐p- ☐p+ ☒c 3/3

Domanda 33 Considera un sistema realtime con i seguenti 2 task periodici:

- A, tempo di esecuzione 3, periodicità 7
- B, tempo di esecuzione 1, periodicità 5

	A	B
×	3	1
P	7	5
PRIO	$\frac{1}{7}$	$\frac{1}{5}$

$u = \frac{3}{7} + \frac{1}{5} \simeq 62,8\%$

$2(2^{\frac{1}{2}} - 1) \simeq 82,8\%$

Scegli l'algoritmo **più semplice/opportuno** tra i due visti durante il corso (RMA o EDF).
► **Indica chiaramente l'algoritmo scelto sulla riga tratteggiata sotto alla tabella e motiva la tua scelta su un foglio a parte.**
► **Completa la tabella indicando l'ordine di scheduling in base all'algoritmo scelto.** *Nota: considera che si tratta di un sistema non-preemptive con una sola CPU/un solo core.* **[3 punti]**

ORDINE: B, A

Tempo (ms) →	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
A	X	X	X					X	X	X					X	X	X					X	X	X					X	X	X		
B	X					X					X					X					X					X					X		
RMA	B	A	A	A		B		A	A	A	B				A	A	A	B			B	A	A	A		B			A	A	A	B	

62,8% < 82,8% => PER SEMPLICITÀ USO RMA ☐w ☐p- ☐p+ ☒c 3/3

Domanda 34 Considera un sistema realtime con i seguenti 3 task periodici:

- A, tempo di esecuzione 1, periodicità 4
- B, tempo di esecuzione 2, periodicità 6
- C, tempo di esecuzione 4, periodicità 10

	A	B	C
X	1	2	4
P	4	6	10

$U = \frac{1}{4} + \frac{2}{6} + \frac{4}{10} \cong 98,3\%$ $3(2^{\frac{1}{3}}-1) \cong 77,9\%$

Scegli l'algoritmo **più semplice/opportuno** tra i due visti durante il corso (RMA o EDF).

► **Indica chiaramente l'algoritmo scelto sulla riga tratteggiata sotto alla tabella e motiva la tua scelta su un foglio a parte.**

► **Completa la tabella indicando l'ordine di scheduling in base all'algoritmo scelto.** *Nota: considera che si tratta di un sistema non-preemptive con una sola CPU/un solo core.* **[3 punti]**

Tempo (ms) →	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
A	X				X				X				X				X				X				X				X				X
B	X	X					X	X					X	X					X	X					X	X					X	X	
C	X	X	X	X							X	X	X	X							X	X	X	X							X	X	X
EDF	A	B	B	C	C	C	C	A	B	B	A	C	C	C	C	A	B	B	A	B	B	A	C	C	C	C	A	B	B	A	B	B	A

98,3% > 77,9% => DEVO USARE EDF

☐w ☐p- ☐p+ ☒c

3/3