Cognome:_____ **Nome:** _____ **Matricola:**_____

Sistemi Operativi — A.A. 2007-2008, prova scritta dell'8 luglio 2008.

Libri e appunti chiusi. Vietato comunicare con chiunque. Vietato l'uso di cellulari, calcolatrici, palmari e affini. Tempo a disposizione: 60 minuti.

1. Considera un sistema con architettura del kernel "execution within user process". In tale sistema sono presenti tre processi: **A**, **B**, **C**, inizialmente tutti e tre in coda ready nell'ordine **A** in testa poi **B** poi **C**. La politica di scheduling è **round robin** con quanto q.
A è cpu bound e genera sempre un page fault dopo 0.5q, il page fault viene servito in tempo 0.3q.
B è I/O bound (cpu burst trascurabili, i/o servito in 0.7q).
C è puramente cpu bound e non provoca page faults.
Il processore esegue di volta in volta A, B, C, e inoltre, con tempi trascurabili, il codice per mode switching, dispatching, system call e interrupt handlers. Mostra schematicamente, nella seguente tabella, l'ordine con cui tali attività vengono eseguite (una sola croce per ciascuna colonna). Indica anche quali processi sono running, quali ready e quali bloccati in ciascun istante come indicato nell'esempio.

tempo →

proc. in user mode	A	x																						x						
	B						x																							
	C										x				x				x											
mode switch			x			x		x			x		x		x		x		x		x				x		x			
kernel mode	dispatching				x					x													x							
	system call i/o								x																		x			
	interrup handler per page fault			x																										
	interrup handler per i/o												x				x													
	interrup handler per q scaduto																			x										
stati processi	running	A	A	A		B	B	B			C	C	C	C	C	C	C	C	C	C		A	A	A	A	A				
	ready	BC	BC	BC	BC	C	C	C	C	C				A	A	A	A	AB	AB	AB	AB	ABC	BC	BC	BC	BC	BC			
	in blocco				A	A	A	A	AB	AB	AB	AB	AB	B	B	B	B														

2. Rispondi brevemente alle seguenti domande sulle politiche di page replacement.
Che significa page buffering?

Tecnica che mantiene temporaneamente in memoria le pagine che sono tolte dal resident set in modo da soddisfare dalla memoria (anziché dal disco) page fault dovuti a scelte non ottimali della politica di page replacement.

Il page buffer è composto di due insiemi: pagine modificate e pagine non modificate.
Le prima devono essere scritte su disco prima di essere riutilizzate, le seconde possono essere riutilizzate subito.

Sistemi Operativi — A.A. 2007-2008, prova scritta dell'8 luglio 2008.

Che significa “minor page fault” in questo contesto?

E' un page fault che non genera un accesso al disco poiché la pagina richiesta e' nel page buffer.

Fuori da contesto del page buffering: si chiamano minor page fault anche quelli che derivano dal fatto che la pagina è parte di una nuova allocazione di memoria e quindi non vi e' nulla da caricare da disco (ad esempio, in unix, system call *brk* e *mmap*).

Elenca i vantaggi di page buffering per la gestione delle pagine modificate?

- le pagine modificate possono essere scritte quando il disco e' idle
- le pagine modificate possono essere scritte sfruttando request merging
- se la pagina viene richiesta dal processo prima di essere scritta si risparmia la scrittura.

3. Considera una architettura Pentium-like: pagina di 4 KB, paginazione a due livelli, pte 4 byte, root page table sempre in memoria. Il frammento di codice assembly mostrato è composto da 2 istruzioni che vengono eseguite consecutivamente. All'inizio dell'esecuzione la prima locazione libera puntata dallo stack pointer è 0xddaff001 e lo stack cresce **verso il basso**, **cioè verso indirizzi minori**. Calcola quanti page fault può generare al più ciascuna istruzione durante l'esecuzione del frammento in questione nelle fasi di fetch e di esecuzione. Considera le istruzioni eseguite di seguito e supponi che le pagine caricate dalle istruzioni precedenti permangano residenti durante l'esecuzione delle istruzioni successive.

Indirizzo	lughezza	istruzione	Page faults dovuti a parti di page table non residenti		Page faults dovuti a codice o dati non residenti	
			fetch	execute	fetch	execute
`0x11bffffd`	`5`	carica nel registro A 4 byte a partire da `0xddaffffe`	2	1	2	2
`0x11c0000`**`2`**	`1`	push del contenuto di A nello stack	0	0	0	1

Supponi che il TLB sia inizialmente vuoto, cosa conterrà dopo l'esecuzione di tali istruzioni?

Il tlb conterra' le page table entry per le seguenti pagine: 11bff, 11c00, ddafe, ddaff, ddb00 (nota, una entry per ciascun possibile “page fault dovuto a codice o dati non residenti”). L'architettura potrebbe inoltre inserire nel TLB anche le entry della root page table per tali pagine, e cioe' 11[10], 11[11], dd[10], gli 8 bit a sinistra sono espressi in esadecimale i due bit a destra, tra quadre, in binario.
(nota, una entry per ciascun possibile “page fault dovuto a parti di page table non residenti”).

Cognome:_____ **Nome:** _____ **Matricola:**_____

Sistemi Operativi — A.A. 2007-2008, prova scritta dell'8 luglio 2008.

Come viene trattato il TLB in caso di process switch?

Viene cancellato.

4. Descrivi l'algoritmo di **disk scheduling** "elevetor".

Vedi materiale didattico.

Rispetto a quale parametro elevator è unfair? Perché? C'è modo di rendere elevator fair?

Il parametro in questione è il "tempo di attesa massimo".

Supponi le tracce numerate da 0 a N. Supponi che arrivi una richiesta per la traccia t (con $0 \leq t \leq N$). Senza perdita di generalità supponi che $t < N/2$ (il caso $t > N/2$ è analogo).
Il caso peggiore si ha quando la testina è sulla traccia t+1 e si sta muovendo verso la traccia N. Il numero di tracce che la testina deve percorrere in questo caso è dato da 2(N-t), quindi il *tempo di attesa massimo* è massimo per t=0 e minimo per $t=\lfloor N/2 \rfloor$ e per $t=\lceil N/2 \rceil$.

La versione "fair" di elevator è one-way elevator ed effettua gli accessi a disco solo quando il movimento è in una delle due direzioni, nell'altra direzione va direttamente all'inizio. Il tempo di attesa massimo per quest'approccio è lo stesso per qualsiasi richiesta.

Perché viene introdotto la tecnica del "**request merging**" e quali sono i vantaggi?

vedi materiale didattico