Domande compiti sistemi operativi:

1. Spiegare il concetto di anomalia di Belady

   L'anomalia di Belady consiste nel fatto che con alcuni algoritmi di sostituzione delle pagine di memoria (ad esempio l'algoritmo FIFO), aumentando il numero di frame può aumentare la frequenza di page fault contrariamente a quanto è ragionevole pensare dal momento che si ha più memoria a disposizione di un processo.L'anomalia di Belady è presente in tutti gli algoritmi che non sono a stack.

2. Spiegare il concetto di thrashing

   E' l'intensa attività di paginazione che avviene quando un processo non ha frame sufficienti e ha un grande numero di pagine in uso attivo. Nel momento in cui i processo necessita di una pagina che non è presente in memoria provoca un page-fault e deve sostituire una delle pagine che però è in uso attivo e quindi sarà subito necessaria e il processo provocherà un nuovo page fault. Quindi il processo continua a emettere fault, sostituendo pagine per le quali emetterà fault che riprenderà immediatamente. Un processo in thrashing spende più tempo per la paginazione che per l'esecuzione. Per evitare il verificarsi del thrashing, occorre fornire ad un processo quanti frame necessita. Esistono varie tecniche per sapere quanti frame "servono" a un processo. Una strategia è quella del working set.

3. Spiegare il concetto di thread

   Un thread è l'unità base di utilizzo della CPU e consiste di un program counter di uno stack e di un insieme di registri. Un insieme di thread e dell'insieme da essi condiviso (sezioni codice e dati e risorse fornite dal sistema operativo come file aperti e segnali) prende il nome di task. Un processo tradizionale è un task con un unico thread. L'alto grado di condivisione fa sì che la creazione dei thread e il passaggio del controllo della CPU da un thread all'altro all'interno di uno stesso task siano meno costose dei context switch tra processi tradizionali perché non implica alcuna operazione legata alla gesione della memoria, mentre nella realizzazione con processi multipli, ciascun processo esegue lo stesso codice ma ha le proprie risorse di memoria e i propri file aperti. Un processo con thread multipli impiega minori risorse-memoria, file aperti, scheduling della CPU- di più processi ridondanti.

4. Spiegare il meccanismo di calcolo del CPU burst in SJF

   La formula che descrive il CPU burst in SJF è la seguente:

   $\tau_{n+1}=\alpha t_n+(1-\alpha)\tau_n$ con $0\leq\alpha\leq1$

   dove t_n è la lunghezza del CPU burst n-esimo e τ_{n+1} è invece il valore previsto del CPU burst successivo, quindi t_n memorizza le informazioni più recenti mentre τ_n memorizza la storia passata. Il parametro α controlla il peso relativo della storia passata e di quella recente. Se il valore di $\alpha=0$, si ha $\tau_{n+1}=\tau_n$ cioè la storia recente non ha effetto, cioè vine supposto che le condizioni attuali siano transitorie. Se $\alpha=1$ invece si ha $\tau_{n+1}=t_n$ cioè ha significato solo il CPU burst più recente. Se invece $\alpha=0,5$ ciò significa che la storia passata e i dati recenti hanno lo stesso peso. Sviluppando in serie la formula si ottiene

   $\tau_{n+1}=\alpha t_n+(1-\alpha)\alpha t_{n-1}+\ldots+(1-\alpha)^j\alpha t_{n-j}+\ldots+(1-\alpha)^{n+1}\tau_0$

   dove, poiché sia α che $1-\alpha\leq1$ ciascun termine ha peso inferiore a quello del suo predecessore

5. Spiegare il meccanismo dell'allocazione indicizzata in UNIX

Molti file vengono memorizzati su uno stesso disco. Il problema principale consiste nell'allocare lo spazio per questi file in modo che lo sapzio sul disco venga utilizzato in modo efficiente e l'accesso al file sia rapido. Esistono tre metodi principali di allocazione: contigua, concatenata o indicizzata. Nell'allocazione indicizzata, ogni file ha un proprio blocco indice cioè un vettore di indirizzi dei blocchi che del disco. L'i-esimo elemento del blocco indice punta sull'i-esimo blocco del file. La directory contiene l'indirizzo del blocco indice. In questo modo ogni file ha un blocco indice e quindi è auspicabile che questo blocco indice sia il più piccolo possibile, ma se è troppo piccolo non è in grado di contenere puntatori sufficienti per un file di grandi dimensioni. Questa situazione può essere gestita in tre modi:

a) Schema concatenato: un blocco indice è formato da un solo blocco del disco e quindi per permettere la presenza di grossi file è possibile collegare tra loro parecchi blocchi indice attraverso puntatori.

b) Indice Multilivello: si ha un blocco indice di primo livello che punta a un insieme di blocchi indice di secondo livello che a loro volta puntano ai blocchi dei file stessi. Per accedere a un blocco il SO utilizza l'indice di primo livello con il quale individua il blocco indice di secondo livello e con esso trova il blocco dati richiesto.

c) Schema combinato: si tengono i primi 15 puntatori del blocco indice nel blocco indice del file. La directory punta all'inode. I primi 12 di questi puntatori contengono direttamente gli indirizzi di blocchi che contengono dati del file. Il 13 puntatore è l'indirizzo di un blocco indiretto singolo: questo è un blocco indice che non contiene dati ma indirizzi di blocchi che contengono dati. Il 14 puntatore è l'indirizzo di un blocco indiretto doppio: questo è un blocco indice che contiene l'indirizzo di un blocco che a sua volta contiene gli indirizzi a di blocchi contenenti i puntatori agli effettivi blocchi di dati. Il 15 puntatore è un puntatore a un blocco indiretto triplo.

d) Spiegare il funzionamento di DMA

e) DMA: Direct Memory Access. Standard hardware definito da un controller (insieme di chip) per effettuare il trasferimento diretto di blocchi di dati su bus, da una periferica come l'hard disk verso la memoria principale, senza coinvolgere la CPU. Il controller DMA dispone di un buffer per poter memorizzare questi dati prima di impossessarsi del bus. Durante il trasferimento, il bus viene occupato dal controller DMA e la CPU può eseguire solo operazioni interne con la cache. La CPU non può accedere alla memoria principale insieme al controller DMA, a meno di un controllo del bus in alternanza.Finito il trasferimento, il DMA invia un interrupt alla CPU che provvederà a controllare se il trasferimento è stato completato con successo. In questo modo si aumentano le prestazioni della macchina poiché mentre la CPU opera con la cache, anche altre periferiche (floppy disk, hard disk, scheda audio) possono disporre del controller DMA per le operazioni di I/O con la memoria. Vediamo i passi di un trasferimento DMA:

   1. Viene chiesto al driver di un dispositivo di trasferire dati dal disco al buffer di memoria di indirizzo X
   2. Il driver chiede al controller del disco di trasferire C byte dal disco al buffer di indirizzo X
   3. Il controller del disco avvia il trasferimento DMA
   4. Il controller del disco invia i dati byte per byte al controller DMA
   5. Il controller DMA trasferisce i byte al buffer X incrementando l'indirizzo di memoria e decrementando C finchè C=0
   6. Quando C=0, il controller DMA genera un interrupt per segnalare alla CPU che il trasferimento è terminato

f) Spiegare le condizioni necessarie affinchè si verifichi un deadlock

a) Mutua esclusione: almeno una risorsa deve essere non condivisibile cioè può essere utilizzata da un solo processo per volta

b) Possesso e attesa: Deve esistere un processo che possedendo almeno una risorsa attende di acquisire risorse possedute da altri processi

c) Impossibilità di prelazione: Una risorsa può essere rilasciata da un processo solo dopo aver terminato il proprio compito.

d) Attesa circolare: deve esistere un gruppo di processi P0,P1,P2,..Pn per cui Pn è in attesa di una risorsa posseduta da P1, P1 è in attesa di una risorsa posseduta da P2, P2 è in attesa di una risorsa posseduta da P3 e così via.

6. Illustrate, anche con un esempio, i concetti di scheduling preemptive e di scheduling cooperativo

Uno scheduler si dice cooperativo se il controllo di una risorsa viene trasferito da un processo ad un altro solo se l'assegnatario attuale lo cede volontariamente

Uno sc heduler si dice preemptive se è possibile che il controllo di una risorsa venga tolto all'assegnatario attuale a causa di un evento.

7. Illustrare il concetto di algoritmo a stack. Portare esempi di algoritmi a stack e algoritmi non a stack.

Un algoritmo di rimpiazzamento delle pagine di memoria si dice a stack se l'insieme delle pagine in memoria per n frame è un sottoinsieme dell'insieme delle pagine che dovrebbero essere in memoria per n+1 frame. L'algoritmo LRU è un algoritmo a stack mentre l'algoritmo FIFO non è un algoritmo a stack.

8. Descrivete concisamente l'algortmo di scheduling SJF, inclusa la formula per il calcolo del CPU burst.

L'algoritmo di scheduling SJF associa ad ogni processo la lunghezza del suo CPU burst successivo. Quando la CPU è disponibile, viene assegnata al processo che ha il CPU burst successivo più breve.

Si tratta di un algoritmo ottimale nel senso che rende minimo il tempo medio di attesa per un dato insieme di processi. Viene spesso utilizzato nello scheduling a lungo termine. Non può essere implementato come scheduling a breve termine poiché non è possibile conosccere la lunghezza del CPU burst successivo. Un possibile approccio è quello di approssimare lo scheduling SJF tentando di predire il valore del prossimo CPU burst. ll CPU burst successivo è generalmente ottenuto effettuando la media esponenziale delle lunghezze misurate dei CPU precedenti. L'algoritmo SJF può essere sia preemptive che cooperativo.

9. Descrivere il concetto di working set

Si tratta di una strategia per evitare il thrasching che consiste nel cercare di sapere quanti frame "servono" a un processo. L'idea del modello del working set è quella di esaminare i piu recenti Δ riferimenti alle pagine. L'insieme delle pagine nei più recenti Δ riferimenti è il working set che appunto è una approssimazione della località del programma. La precisione del working set dipende dalla scelta di Δ. Se è troppo grande rischia di sovrapporre più località, se è troppo piccolo rischie di non includere l'intera località. La caratteristica più importante del working set è appunto la sua dimensione. Calcolando la dimensione del working set per ogni processo del sistema è possibile determinare la richiesta totale di frame. Se la richiesta totale è maggiore di del numero totale di frame liberi, si verifica thrashing in quanto alcuni processi non dispongono di frame a sufficienza. Il sistema operativo monitorizza il working set di ogni processo e gli alloca un numero di frame sufficienti per fornirgli la sua dimensione del working set. Se la dimensione del working set aumenta superando il numero totale di frame disponibili, il sistema operativo seleziona un processo da sospendere. La difficoltà insita nel modello del working set consiste nel tenere traccia del working set stesso poiché si tratta di un insieme in movimento.

10. Descrivere le tecniche per verificare la coerenza di un file system

11. Descrivere concisamente il concetto di MFT di Windows

12. Descrivere i principali meccanismi per la realizzazione di directory basate su grafi aciclici

La struttura ad albero proibisce la condivisione di file o directory. Un grafo aciclico invece permette alle directory di avere sottodirectory e file condivisi, cioè lo stesso file o la stessa directory possono trovarsi in due directory diverse. I file e le directory condivise possono essere implementate in molti modi:

a) Duplicazione di tutte le informazioni relative ai file in entrambe le directory condivise: la copia e l'originale vengono ad essere indistinguibili, sorge il problema di mantenere la coerenza se il file viene modificato.

b) Link simbolici: viene crato un tipo speciale di directory entry che contiene un riferimento sotto forma di cammino assoluto al file in questione. Nel caso di cancellazione di un link ciò non influisce sul file originale. Se viene cancellato il file, lo spazio corrispondente viene deallocato lasciando in sospeso il link; a questo punto si dovrebbe cercare tutti questi link e cancellarli ma se in ogni file non esiste una lista dei link associati, la ricerca può risultare piuttosto costosa. In alternativa un link può essere lasciato finchè non viene fatto un tentativo di riutilizzarli. In questo caso il progettista deve decidere cosa fare quando un file viene cancellato e viene creato un altro file con lo stesso nome, prima che venga usato un link simbolico al file originale. In UNIX quando un file è viene cancellato i link simbolici continuano ad esistere ed è l'utente che deve rendersi conto che il file originale è scomparso.

c) Hard link: le informazioni relative ad un file vengono copiate in entrambe le directory condivise. Sono di fatto una copia di una voce di directory, hanno nomi diversi ma puntano allo stesso inode e quindi condividono esattamente lo stesso dato (oltre agli stessi permessi, data di modifica, owner ecc.).Si deve utilizzare la tecnica degli inode che contengono anche un contatore di riferimenti. Un nuovo link o un nuovo elemento della directoru aumenta il numero di riferimenti. Cancellando un link o un elemento questo numero si riduce. Quando il contatore è uguale a zero il file può essere cancellato.

13. Spiegare il concetto di Memory Mapped I/O

Nel caso di operazioni di I/O, ciascun controller di I/O (di una periferica si distinguono solitamente la parte elettronica, denominata controller (o processore di I/O), e quella meccanica chiamata dispositivo. Il controller è il circuito elettronico di input/output, che opera da interfaccia tra CPU e uno o più dispositivi meccanici o elettrici) è dotato di registri nei quali mantenere i comandi e i dati da trasferire. Solitamente speciali istruzioni di I/O permettono il trasferimento dati tra questi registri e la memoria centrale. Per rendere più conveniente l'accesso ai dispositivi di I/O molte architetture forniscono una particolare tecnica di I/O che prende il nome di I/O mappato in memoria e che consiste nel riservare un certo insieme di indirizzi di memoria da associare ai registri dei dispositivi. In questo modo ogni operazione di lettura o scrittura a questi indirizzi di memoria comporta un trasferimento diretto di dati con i registi del dispositivo. L'uso del I/O mappato in memoria è utile per esempio nei dispositivi come le porte seriale o parallele usate per collegare ai computer modem e stampanti. La CPU trasferisce i dati attraverso questi tipi di dispositivi (detti porte di I/O) leggendo e scrivendo in alcuni registi in essi contenuti. Per inviara dati in una porta mappata in memoria, la CPU scrive un bute nel registro dei dati, quindi imposta un bit nel registro di controllo per segnalare che il byte è disponibile. Il dispositivo riceve il byte e quindi cancella il bit del registro di controllo per segnalare che è disponibile a ricevere un nuovo byte.

14. Spiegare l'algoritmo Rate Monotonic

E' una politica di scheduling valida se:

a) Ogni processo periodico deve completare entro il suo periodo

b) Tutti i processi sono indipendenti

c) La preemption avviene istantaneamente e senza overhead

Viene assegnata staticamente una priorità ad ogni processo

Processi con frequenza più alta hanno priorità più alta. Ad ogni istante viene eseguito il processo con priorità più alta (facendo preemption).

15. Spiegare l'algoritmo EDF

    E' una politica di scheduling per processi periodici real time. Viene scelto di volta in volta il processo che ha la deadline più prossima. Viene detto a priorità dinamica perché la piorità relativa di due processi varia in momenti diversi.

16. Spiegare il concetto di loading e linking dinamici

    Il caricamento dinamico consiste nel fatto che una routine viene caricata solo quando viene richiamata. Non è necessario supporto speciale da parte del SO.

    Il collegamento dinamico è analogo al caricamento, solo che anziché posticipare il caricamento di una routine fino al momento dell'esecuzione, viene posticipato il suo collegamento. Questa caratteristica viene utilizzata con librerie di sistema. Possono esservi problemi di versioning.

17. Spiegare il concetto di File Allocation Table

    Si tratta di una variante del metodo di allocazione concatenata utilizzata dai sistemi operativi DOS e OS/2. Per contenere tale tabella viene riservata una sezione del disco all'inizio di ciascuna partizione; la FAT ha un elemento per ogni blocco e viene indicizzata dal numero del blocco. Essa viene utilizzata essenzialmente come una lista concatenata. L'elemento di directory contiene il numero del primo blocco del file. L'elemento della tabella indicizzato da quel numero di blocco contiene a sua volta il numero di blocco del blocco successivo del file. Questo schema può causare un notevole numero di posizionamenti della testina. Un vantaggio è dato dall'ottimizzazione del tempo di accesso casuale.

18. Spiegare il concetto di disk striping in RAID

    Per aumentare la velocità di un componente, una delle possibilità è quella di utilizzare il parallelismo. L'idea è quella di utilizzare un array di dischi indipendenti, che possano gestire più richieste di I/O in parallelo. Dobbiamo però garantire che i dati letti in parallelo risiedano su dischi indipendenti. Lo striping non dovrebbe essere un membro a tutti gli effetti della famiglia RAID perché non possiede meccanismi di ridondanza. I dati vengono distribuiti su più dischi cosicchè se due richieste di I/O riguardano blocchi indipendenti, c'è la possibilità che i blocchi siano su dischi differenti cosicchè le due richieste possono essere servite in parallelo. I dati vengono suddivisi in strip (settori, blocchi o altro multiplo). Strip consecutivi sono distribuiti su dischi diversi, aumentando le performance della lettura dei dati sequenziali. Vi è efficacia sia per grandi trasferimenti di dati se la quantità di dati richiesta è grande rispetto alla dimensione degli strip e per un gran numero di richieste indipendenti se la quantità di dati richiesta è paragonabile alla dimensione degli strip.

19. Spiegare un meccanismo a scelta di deadlock prevention

    Si elimina una delle quattro condizioni. Il deadlock viene eliminato strutturalmente

    a) Attaccare la condizione di mutua esclusione permettendo la condivisione di risorse. Ad esempio con uno spool di stampa tutti i processi pensano di usare contemporaneamente la stampante. Non è sempre possibile applicare lo spooling. Inoltre si sposta il problema verso altre risorse.

    b) Attaccare la condizione di possesso e attesa : due meccanismi sono possibili:

        - Ogni processo richiede tutte le risorse dall'inizio (non sempre l'insieme delle richieste è noto fin dall'inizio e inoltre è inefficiente perché molte risorse saranno allocate ma magari non utilizzate per un lungo periodo di tempo);

        - Un processo può richiedere risorse solo se non ne possiede o deve prima rilasciare le risorse che possiede

    Questi protocolli possono portare a starvation.

c) Attaccare la condizione di assenza di prerilascio

   Se un processo che possiede risorse ne richiede una che non può essergli allocata subito, allora tutte le risorse in suo possesso vengono prelazionate . Quindi queste risorse sono rilasciate implicitamente.

d) Attaccare la condizione di attesa circolare: Ad ogni tipo di risorsa viene assegnato un nmero intero che permette di confrontare due risorse e stabilire un ordinamento. Ogni processo può richidere risorse solo seguento un ordine crescente di numerazione o, in alternativa che un processo prima di richiedere un'istanza di un tipo di risorsa, rilasci qualsiasi risorsa con numero maggiore o uguale.

20. Spiegare un meccanismo a scelta per l'allocazione di memoria secondaria

    Esistono tre metodi principali per l'allocazione dello spazio su disco:

a) Allocazione contigua: ogni file deve occupare un insieme di blocchi contigui del disco. E' facile accedere ad un file allocato in modo contiguo. Una difficoltà riguarda l'individuazione dello spazio per un nuovo file. Altro problema è la determinazione della quantità di spazio necessaria per un file. Quando il file viene creato, occorre trovare e allcoarae lo spazio di cui necessita. Esiste il problema di conoscere la dimensione del file da creare perché se a un file viene allocato poco spazio, può essere impossibile estendere il file.

b) Allocazione concatenata: Ogni file è costituito da una lista contcatenata di blocchi del disco che possono essere sparsi in qualsiasi punto del disco stesso. La directory contiene un puntatore al primo e all'ultimo blocco del file. Ogni blocco contiene un puntatore a blocco successivo. Con l'allocazione concatenata non esiste frammentazione esterna. Può però essere utilizzata solo per file con accesso sequenziale. Un altro svantaggio riguarda lo spazio richiesto per i puntatori. La soluzione comune consiste nel raccogliere i blocchi in gruppi detti cluster e nell'allocare i cluster anziché i blocchi. Altro problema riguarda l'affidabilità dovuta al possibile danneggiamento di un puntatore.

c) Allocazione indicizzata: ogni file ha un proprio blocco indice: si tratta di un vettore di indirizzi ai blocchi del disco. L'i-esimo elemento del blocco indice punta all'iesimo blocco del file. La directory contiene l'indirizzo del blocco indice. L'allocazione indicizzata supporta l'accesso diretto senza soffrire di frammentazione esterna. Con l'allocazione indicizzata si ha un certo spreco di spazio dovuto alla dimensione dei puntatori che è maggiore che nel caso dell'allocazione concatenata. Vi è inoltre il problema della dimensione del blocco indice. Tre schemi: concatenato, con indice multilivello e schema combinato.

21. Spiegare un meccanismo a scelta di deadlock avoidance

    Prima di assegnare una risorsa ad un processo, si controlla se l'operazione può portare al pericolo di deadlock. In quest'ultimo caso l'operazione viene ritardata.

    Algoritmo del banchiere.

22. Spiegare il concetto di binding

    Il binding è l'associazione di istruzioni e dati a indirizzi di memoria. Può avvenire sia in fase di compilazione che di caricamento che di esecuzione (con hardware speciale MMU). Se avviene durante la compilazione gli indirizzi vengono calcolati al momento della compilazione e resteranno gli stessi ad ogni esecuzione del programma. E' semplice e veloce. Se avviene durante il caricamento il codice contiene indirizzi relativi (codice rilocabile). In questo modo è possibile gestire la multiprogrammazione.

23. Spiegare un meccanismo a scelta per aumentare l'affidabilità dei sistemi di memorizzazione in memoria secondaria

24. Illustrate l'organizzazione di un File sistem di tipo Unix (ad esempio EXT2)

25. Spiegare quali sono gli stati di un processo UNIX e le condizioni che determinano le varie transizioni di stato

26. Descrivere brevemente i problemi legati ai deadlock e come modellarli

    In un ambiente multiprogrammato più processi possono entrare in competizione per ottenere un numero finito di risorse. Un processo richiede una risorsa e se in quel momento essa non è disponibile il processo viene messo in condizione di attesa. Può succedere che un processo in attesa non cambi più il suo stato; ciò avviene quando le risorse richieste vengono trattenute da altri processi in attesa. Situazioni simili sono chiamate deadlock. Un gruppo di processi entra in deadlock quando tutti i processi del gruppo attendono un evento che pupò essere causato solo da un altro processo in attesa.

27. Illustrare il funzionamento di un sistema RAID

    Per aumentare la velocità di un componente, una delle possibilità è quella di utilizzare il parallelismo. L'idea è quella di utilizzare un array di dischi indipendenti, che possano gestire più richieste di I/O in parallelo. Dobbiamo però garantire che i dati letti in parallelo risiedano su dischi indipendenti. RAID è uno standard per l'utilizzo di più dischi in parallelo. Questo array di dischi viene visto dal SO come un singolo disco logico. I dati sono distribuiti fra i vari dischi dell'array. La capacità ridondante dei dischi può essere utilizzata per memorizzare informazioni di parità che garantiscono il recovery dei dati in caso di guasti. L'utilizzo di più dischi aumenta le probabilità di guasto del sistema. Per compensare la riduzione di affidabilità si utilizzano meccanismi di parità.

28. Descrivere concisamente il concetto di aging

    Si tratta di una tecnica che ha come scopo quello di evitare che un processo sia soggetto a starvation in uno schedule a priorità. Questa tecnica consiste nell'incrementare gradualmente la priorità dei processi in attesa. Un processo quindi non può restare in attesa per un tempo indefinito perché prima o poi raggiungerà la priorità massima.

29. Descrivere concisamente il concetto di microkernel

    Microkernel: Idea di rimuovere dal kernel tutte le parti non essenziali e implementarle come processi a livello utente. Semplici astrazioni dell'HW gestite e coordinate da un kernel minimale basate su un paradigma client/server e primitive di message passing. Il kernel risultante è molto semplice e più facilmente espandibile e modificabile e più facilmente portabile ad altre architetture. Più sicuro perché è possibile assegnare al microkernel e ai processi di sistema livelli di sicurezza diversi. Vi è una maggiore inefficienza dovuta all'overhead determinato dalla comunicazione mediata tramite kernel del sistema operativo.

30. Descrivere concisamente le tecniche di allocazione dinamica first fit, next fit, best fit,worst fit

    Allocazione dinamica della memoria principale: durante l'esecuzione un programma può essere spostato all'interno della memoria. Abbiamo bisogno di una struttura dati per mantenere informazioni sulle zone libere e su quelle occupate come mappe di bit o liste con puntatori. Indipendentemente dalla struttura dati utilizzata, l'operazione di selezione di un blocco libero può avvenire con i seguenti algoritmi:

    a) First fit: scorrimento della lista di blocchi liberi fino a trovare il primo vuoto grande abbastanza per contenere il processo

    b) Next fit: come first solo che parte da dove si era fermato in precedenza (studiato per evitare la frammentazione della prima parte ma ha performance peggiori di First Fit)

    c) Best fit: Seleziona il più piccolo fra i blocchi liberi (genera più frammentazione di FF e riempie la memoria di blocchi liberi piccoli)

d) Worst fit: seleziona il più grande tra i blocchi liberi (rende difficile l'allocazione di processi di grosse dimensioni)

31. Descrivere concisamente il concetto di journaled file system (file system basato su log)

    Transazione: ogni aggiornamento del file system. Tutte le transazioni vengono memorizzate su un log. Periodicamente le transazioni nel log vengono effettuate nel file system e la transazione viene rimossa dal log. Se il sistema i guasta tutte le transazioni presenti nel log devono essere ripetute.

32. Descrivere consisamente il supporto hardware per la paginazione

    Memoria fisica suddivisa in blocchi di dimensioni fisse detti frame, la memoria logica è divisa in blocchi di uguale dimensione detti pagine. Ogni indirizzo generato dalla CPU è diviso in due parti un numero di pagine e un offset di pagina. Il numero di pagina serve come indice per la tabella delle pagine che contiene l'indirizzo base di ogni pagina nella memoria fisica. Questo indirizzo viene associato all'offset di pagina per definire l'indirizzo di memoria fisica.Gli indirizzi logici sono mappati in indirizzi fisici. Questo mapping non è visibile da parte dell'utente ed è controllato dal SO che mantiene aggiornata una struttura che si chiama tabella dei frame. Il SO conserva anche una copia della tabella delle pagine per ciascun processo. La realizzazione hardware della tabella delle pagine può essere realizzata in modi diversi:

    a) registri dedicati (efficace se la tabella è piccola)

    b) registri associativi (TLB). Ogni registro è formato da una chiave e da un valore. I registri associativi contengono una piccola parte della tabella delle pagine. Quando la CPU genera un indirizzo logico, il suo numero di pagina viene prsentato ad un insieme di registri associativi. Se nei registri associativi viene trovato allora si ha il corispondente numero di frame e si può accedere alla memoria. Se il numero di pagina non è presente nei registri associativi, occorre fare riferimento alla tabella delle pagine.

33. Descrivere consisamente il meccanismo di gestione degli interrupt

    L'hardware della CPU include un filo detto linea di richiesta dell'interrupt che viene controllato dalla CPU dopo l'esecuzione di ogni istruzione. Quando viene rilevato il segnale di un controller sulla linea degli interrupt, la CPU salva una piccola quantità di informazioni e salta all'interrupt handler. Questo determina le cause dell'interrupt e porta a termine l'eleaborazione necessaria ed esegue una istruzione di ritorno. Può essere necessario poter differire la gestione dell'interrupt durante elaborazioni critiche (interrupt mascherabili). Può poi essere necessario avere diverse priorità di interrupt.Può essere necessario anche disabilitare gli interrupt durante la gestione degli altri interrupt. Inoltre si ha una struttura detta interrupt vector i cui elementi sono gli indirizzi di memoria dei gestori specializzati degli interrupt cosicchè non è necessario che un singolo gestore debba individuare tutte le possibili fonti di interrupt per determinare quale di esse abbia richiesto un servizio. Poiché vi sono più dispositivi che elementi nel vettore di interrupt, ogni elemento del vettore è un puntatore a una lista di gestori di interrupt.

34. Descrivere il concetto di funzione one-way

    Si tratta di una funzione che dato x è facilmente calcolabile f(x) mentre dato f(x) è computazionalmente difficile o impossibile calcoalare x. Queste funzioni generalmente tendono a mescolare i bit in modo molto complesso con operazioni di bit swapping, inversioni ecc.l Calcolare f(x) consiste nel seguire un algoritmo, invertire l'algoritmo è difficile.

35. Descrivere il concetto di buffer-overflow

    Vulnerabilità di tipo software. L'idea è quella di fornire ad un programma un insieme di dati di dimensioni superiori a quella prevista. Nel migliore dei casi il programma va in crash, nel peggiore è possibile che un attaccante prensa il controllo della macchina. L'idea è quella di riempire il buffer con un codice maligno o con un nuovo indirizzo di ritorno che punti a tale codice maligno. Le contromisure possono essere dei patch per compilatori che aggiungono controlli sulla dimensione del buffer o che controllano l'indirizzo di ritorno.

36. Illustrare il meccanismo di gestione delle password basate su salt

37. Descrivere il concetto di matrice di accesso

38. Descrivere il concetto di verme

    Programma che diffonde copie di sé stesso in una rete. I worm sono programmi autonomi e non infettano programmi esistenti.

39. Descrivere il concetto di cavallo di troia

    E' un programma che replica le funzionalità dei programmi di uso comune che contengono codice malefico. Catturano informazioni critiche per la sicurezza del sistema o informazioni private dell'utente. Esempi sono i programmi spyware.

40. Descrivere il concetto di capability

41. Pregi e difetti della allocazione concatenata di memoria secondaria

    L'allocazione concatenata non presenta frammentazione esterna. Può essere utilizzata efficientemente solo per file con accesso sequenziale. Perché per trovare l'i-esimo blocco di un file occorre partire dall'inizio del file e seguire i puntatori finchè non si arriva all'i-esimo blocco. Ogni accesso a un puntatore implica una lettura del disco e talvolta un posizionamento della testina del disco. Un altro svantaggio riguarda lo spazio richiesto per i puntatori.

42. Algoritmo dell'ascensore pregi e difetti delle diverse implementazioni

    Con l'algoritmo SCAN, il braccio dell'unità disco parte da un estremo del disco e si sposta nella sola direzione possibile servendo le richieste mentre attraversa i cilindri fino a che non giunge all'altro estremo del disco: a questo punto inverte la marcia e la procedura continua. Le testine attraversano continuamente il disco nelle due direzioni. L'algoritmo C-SCAN è una variante concepita per garantire un tempo di attesa meno variabile. Anche C-SCAN come SCAN sposta la tresina da un estremo all'altro ma quando giunge ad un estremo torna all'inizio ddel disco senza servire richieste durante il viaggio di ritorno. In realtà si usano due versioni che si chiamano LOOK e C-LOOK che spostano il braccio della testina fino a che non vi siano altre richieste da servire senza arrivare fino alle estremità del disco.

43. Cache del File System e algoritmi di minimizzazione del tempo di seek, come interagiscono

44. Allocazione gerarchica delle risorse

    Ad ogni risorsa viene attribuito un valore.

45. Discutere quali effetti comporta sulle funzionalità e l'efficienza di un sistema operativo l'assenza del meccanismo hardware n:

    a) Modalità supervisore

       Per garantire il corretto funzionamento del sistema è necessario proteggere tanto il sistema operativo quanto gli altri programmi unitamente ai loro dati da qualsiasi programma malfunzionante. Questa protezione è realizzata con un supporto hardware che fornisca due modi di funzionamento: modo supervisore e modo utente. Questo meccanismo consente di proteggere il SO dal comportamento degli altri utenti e di proteggere gli utenti stessi dagli atri utenti. Questo livello di protezione è ottenuto definendo le istruzioni di macchina in grado di causare danni allo stato del sistema come istruzioni privilegiate. Se si esegue in modo utente un'istruzione privilegiata, l'hardware non la esegue e la tratta come un'istruzione illegale generando una trap al SO. I processi in modalità supervisore hanno accesso a tutte le istruzioni, incluse quelle privilegiate che permettono di gestire totalmente il sistema.

b) Reference bit

Questo bit consente di stabilire se l'istruzione corrente è eseguita per conto del SO o per conto dell'utente. All'avvio l'hardware è posto in modo monitor. Viene caricato il SO che provvede all'esecuzione dei processi utenti in modo utente. Ogni volta che si verifica un interrupt l'hardware passa in modo monitor (0) e il SO prende il controllo del computer. Prima di passare il controllo al programma utente il sistema ripristina il modo utente.

c) TLB

Si tratta di una cache hardware di ricerca veloce che contiene una parte della tabella delle pagine. Se non si avesse la TLB non la tabella delle pagine dovrebbe risiedere nella memoria principale e quindi occorrerebbero due accessi in memoria per accedere a un byte.

d) DMA

L'uso di una costosa CPU per il controllo dei bit di stato e per la scrittura di dati sul registro del controller un byte alla volta (I/O programmato) sembra essere uno spreco. Molti computer evitano di sovraccaricare la CPU assegnando parte di questi compiti a un processore specializzato detto appunto controller per l'accesso diretto in memoria (DMA).

e) Timer

E' il dispositivo hardware che agisce come sveglia del processore e fornendo un interrupt allo scadere del tempo prefissato permette al SO di schedulare i processi con il metodo RR.

f) Interrupt

Se non vi fosse il meccanismo di gestione dell'I/O per mezzo di interrupt, la gestione delle operazioni di I/O dovrebbe avvenire per mezzo di interrogazione ciclica da parte della CPU del registro status di un dispositiva fino a che il bit busy non assuma il vlore 0 per indicare che il dispositivo è pronto. Questo meccanismo, di per sé efficiente, diventa inefficiente quando si trova raramente un dispositivo pronto per essere servito mentre rimangono in attesa altre utili elaborazioni richieste dalla CPU. In questi casi è molto più conveniente che il controller notifichi alla CPU che il dispositivo è pronto. Tale meccanismo hardware è l'interrupt.

46. Descrivere uno dei seguenti metodi ponendo in evidenza limiti, vantaggi e svantaggi ripstto a tutti gli altri metodi

a) Deadlock avoidance

Si richiede che al sistema operativo siano date in anticipo ulteriori informazioni relative alle risorse che un processo richiederà e userà durante le sue attività. Con queste informazioni è possibile decidere se una richiesta di risorse da parte di un processo può essere soddisfatta o debba invece essere sospesa.

b) Deadlock detection per mezzo di grafi wait for

Se tutte le risorse hanno una sola istanza è possibile definire un algoritmo di rilevamento di deadlock che fa uso di una variante del grafo di allocazione delle risorse, detta grafo di attesa ottenuta dal grafo di allocazione risorse togliando i nodi delle risorse e avvicinando gli archi tra i processi. Nel sistema esiste un deadlock se e solo se il grafo di attesa contiene un ciclo. Per individuare i deadlock il sistema deve conservare il grafo di attesa e richiamare periodicamente un algoritmo che individui un ciclo all'interno del grafo. Tale algoritmo è dell'ordine di n^2 dove n è il numero di vertici.

c) Deadlock detection per mezzo di grafi di Holt

Un grafo di Holt si dice riducibile se esiste almeno un nodo processo con solo archi entranti. La riduzione consiste nell'eliminare tutti gli archi di tale nodo e riassegnare le risorse ad altri processi. Se le risorse sono ad accesso mutualmente esclusivo seriali e non prerilasciabili, lo stato non è di deadlock se e solo se il grafo di Holt è completamente riducibile cioè se esiste una seguenza di passi di riduzioni che elimina tutti gli archi del grafo.

d) Deadlock prevention

Si elimina una delle quattro condizioni necessarie per il verificarsi di deadlock. Il deadlock viene eliminato strutturalmente.

47. Spiegare in dettaglio quello che accade in risposta a una richiesta di accesso a una pagina non presente in memoria

a) La MMU scopre che una certa pagina non si trova in memoria principale

b) Viene generata una trap di page-fault che viene catturato dal SO

c) Il SO cerca in memoria secondaria la pagina da caricare

d) Il SO individua un frame libero

e) Se non esiste un frame libero:

- Richiama un algoritmo di rimpiazzamento
- Aggiorna la tabella delle pagine
- Se la pagina vittima è stata variata scrive la pagina sul disco
- Aggiorna la tabella dei frame liberi

f) Aggiorna la tabella dei frame (frame occupato)

g) Il SO carica la memoria principale con il contenuto della pagina

h) Il SO aggiorna la page table in modo opportuno e riavvia l'esecuzione

48. Spiegare in dettaglio tutto quello che accade in risposta a una richiesta di scrittura di una parola su un file residente su dusco rigido già aperto in precedenza

49. Spiegare in dettaglio tutto quello che accade quando un processo in ambiente UNIX chiede di aprire in modalità append il file "/mnt/floppy/README.txt". Supporre che in precedenza il filesystem di un floppy dis sia stato mounted tramite il comando "mount/dev/hdb1/mnt"

50. Spiegare in dettaglio la differenza fra thread e processi valutandone pregi e difetti. Soffermarsi sulle problematiche di implementazione dei thread nei sistemi operativi.

Ciascun processo ha una singola "linea di controllo". Per ogni processo viene eseguita una singola sequenza di istruzioni.l Ogni singolo processo non può eseguire due differenti attività contemporaneamente. Thread ha una copia dello stato del processore un proprio Program Counter e uno stack separato. I thread appartenenti allo stesso processo condividono codice, dati, risorse di I/O. E' più economico gestire i thread perhè è più economico fare context switch fra thread piuttosto che fra processi. Si hanno kernel thread che vengono supportati direttamente dal SO e user thread che sono implementati da una libreria a livello utente.

51. Elencare vantaggi e svantaggi discutendo le situazioni reali che portano gli implementatori di sistemi operativi a compiere determinate scelte:

a) Metti a confronto i differenti tipi di architettura per kernel di sistemi operativi

- Kernel monolitici: insieme completo e unico di procedure mutualmente correlate e coordinate. Ottimi livelli di efficienza.

- Microkernel: Idea di rimuovere dal kernel tutte le parti non essenziali e implementarle come processi a livello utente. Semplici astrazioni dell'HW gestite e coordinate da un kernel minimale basate su un paradigma client/server e primitive di message passing. Il kernel risultante è molto semplice e più facilmente espandibile e modificabile e più facilmente portabile ad altre architetture. Più sicuro perché è possibile assegnare al microkernel e ai processi di sistema livelli di sicurezza diversi. Vi è una maggiore inefficienza dovuta all'overhead determinato dalla comunicazione mediata tramite kernel del sistema operativo.

- Kernel ibridi sono simili ai microkernel solo che hanno una parte di codice in "kernel space" per ragioni di maggiore efficienza di esecuzione

- Exokernel : non forniscono livelli di astrazione dell'HW ma forniscono librerie che mettono a contatto diretto le applicazioni con l'HW.

b) Metti a confronto due sistemi utilizzabili nel kernel di un sistema operativo per implementare sezioni critiche

Abbiamo approcci software (semafori, monitor, message passing) con i quali la responsabilità cade sui processi che vogliono accedere ad un oggetto distribuito e presentano il problema del busy waiting e approcci hardware che utilizzano istruzioni speciali del linguaggio progettate apposta. Sono efficienti me non sono adatti come soluzioni general purpose, presentano ancora busy waiting e non eliminano i problemi di starvation.

c) Metti a confronto allocazione concatenata e allocazione indicizzata, due tecniche per l'implementazione di file system

- Allocazione concatenata: Ogni file è costituito da una lista contcatenata di blocchi del disco che possono essere sparsi in qualsiasi punto del disco stesso. La directory contiene un puntatore al primo e all'ultimo blocco del file. Ogni blocco contiene un puntatore a blocco successivo. Con l'allocazione concatenata non esiste frammentazione esterna. Può però essere utilizzata solo per file con accesso sequenziale. Un altro svantaggio riguarda lo spazio richiesto per i puntatori. La soluzione comune consiste nel raccogliere i blocchi in gruppi detti cluster e nell'allocare i cluster anziché i blocchi. Altro problema riguarda l'affidabilità dovuta al possibile danneggiamento di un puntatore.

- Allocazione indicizzata: ogni file ha un proprio blocco indice: si tratta di un vettore di indirizzi ai blocchi del disco. L'i-esimo elemento del blocco indice punta all'iesimo blocco del file. La directory contiene l'indirizzo del blocco indice. L'allocazione indicizzata supporta l'accesso diretto senza soffrire di frammentazione esterna. Con l'allocazione indicizzata si ha un certo spreco di spazio dovuto alla dimensione dei puntatori che è maggiore che nel caso dell'allocazione concatenata. Vi è inoltre il problema della dimensione del blocco indice. Tre schemi: concatenato, con indice multilivello e schema combinato.

d) Metti a confronto first fit, next fit, best fit, worst fit, alcune politiche per l'allocazione di memoria principale

Allocazione dinamica della memoria principale: durante l'esecuzione un programma può essere spostato all'interno della memoria. Abbiamo bisogno di una struttura dati per mantenere informazioni sulle zone libere e su quelle occupate come mappe di bit o liste con puntatori. Indipendentemente dalla struttura dati utilizzata, l'operazione di selezione di un blocco libero può avvenire con i seguenti algoritmi:

- First fit: scorrimento della lista di blocchi liberi fino a trovare il primo vuoto grande abbastanza per contenere il processo
- Next fit: come first solo che parte da dove si era fermato in precedenza (studiato per evitare la frammentazione della prima parte ma ha performance peggiori di First Fit)
- Best fit: Seleziona il più piccolo fra i blocchi liberi (genera più frammentazione di FF e riempie la memoria di blocchi liberi piccoli)
- Worst fit: seleziona il più grande tra i blocchi liberi (rende difficile l'allocazione di processi di grosse dimensioni)

52. Dare la definizione di starvation, mostrando una situazione in cui ci sia starvation. Descrivere un metodo generale per eliminare starvation

E' possibile che un processo non possa accedere ad una risorsa perché "sempre occupata". Ad esempio coda con "furbi". A differenza del deadlock non è una condizione definitiva ma è possibile uscirne adottando una opportuna politica di assegnamento.

53. Elencare tutti gli interrupt e le trap gestiti dai componenti del sistema operativo, illustrando anche le interazioni (dirette o indirette) con l'hardware collegate alla gestione di tali interrupt e trap

a) Memory manager

b) Disk manager

c) Scheduler

d) File system manager