Il file system

Un file è un tipo di dati astratto definito e realizzato dal S.O. Il più semplice metodo d'accesso è quello sequenziale: le informazioni del file si elaborano ordinatamente, un record dopo l'altro.
Un altro è l'accesso diretto: il file si considera come una sequenza numerata di blocchi o record che si possono leggere o scrivere in modo arbitrario.

Struttura di directory

Directory a singolo e due livelli: La struttura più semplice per una directory è quella a singolo livello. Tutti i file sono contenuti nella stessa directory. Nella struttura a due livelli ogni utente dispone della propria directory (UFD) e non si presentano quindi problemi quando ci sono file con lo stesso nome fra i vari utenti.
Lo svantaggio è che non è presente cooperazione e condivisione fra gli utenti.
Directory con struttura ad albero:E' la generalizzazione della struttura a due livelli. Permette di creare proprie sottodirectory e di organizzare i file di conseguenza.
E' presente una directory corrente per ogni utente per fare riferimento ai file. Se non si trovano bisogna specificare un percorso che parte dalla radice o dalla directory corrente.
Directory con struttura a grafo aciclico e generale: Le strutture a grafo aciclico permettono la condivisione di sottodirectory e file, ma complicano le funzioni di ricerca e cancellazione. Una struttura a grafo generale permette la massima flessibilità nella condivisione dei file e delle directory, ma talvolta richiede operazioni di 'ripulitura' (*garbage collection*) per recuperare lo spazio inutilizzato nei dischi.

Montaggio di un file system

I dischi sono suddivisi in una o più partizioni, ciascuna contenente un file sytem o priva di struttura (*raw partition*). Per essere resi disponibili, i file system si possono montare nelle strutture di nominazione del sistema che variano tra loro. Una volta che una partizione è stata montata, i file in essa contenuti sono disponibili per l'uso. I file system si possono smontare per disabilitarne l'accesso o per attività di manutenzione.

Condivisione e protezione

Si sono adottati i concetti di proprietario e gruppo che sono caratterizzati da identificatori (*user*ID e *group*ID) per concedere i permessi per le operazioni richieste.
La condivisione si utilizza anche per file system remoti, dove attraverso specifici programmi si permette il montaggio del modello client-server dove il server dichiara che determinate risorse sono disponibili ai client, specificando esattamente quali risorse ed esattamente a quali client.
(parte dei file system distribuiti).
Lo schema più generale per realizzare gli accessi dipendenti dall'identità consiste nell'associare un elenco di controllo degli accessi a ogni file e directory. Quando un utente richiede un accesso a un file specifico il S.O. esamina l'elenco associato a quel file; se tale utente è presente nell'elenco si autorizza l'accesso.
Per evitare inconvenienti si introduce una versione condensata dell'elenco di controllo dove si raggruppano gli utenti in tre classi distinte: **proprietario**, **gruppo**, **universo**.

Realizzazione delle directory

Il metodo più semplice è basato sull'uso di una lista lineare: l'individuazione di un elemento richiede una ricerca lineare, quindi è di facile programmazione ma la sua esecuzione è onerosa in termini di tempo, in quanto le informazioni sulla directory vengono usate frequentemente e si avvertirebbe quindi una gestione lenta d'accesso.
Un'altra struttura è la tabella hash. Riceve un valore calcolato usando come operando il nome del file e riporta un puntatore al nome del file nella lista lineare. Diminuisce il tempo di ricerca nella directory, anche se bisogna evitare collisioni (due nomi di file con stessa locazione).

Metodi di assegnazione

Assegnazione contigua: Ogni file deve occupare un insieme di blocchi contigui del disco. Quando si usa un accesso sequenziale, il file system memorizza l'indirizzo dell'ultimo blocco cui è stato fatto riferimento e legge il blocco successivo. Nel caso di un accesso diretto al blocco *i* di un file che comincia al blocco *b* si può accedere immediatamente al blocco $b+i$.
Però si presentano alcuni problemi: il problema generale è quello di soddisfare una richiesta di dimensione n data una lista di buchi liberi, e in questo caso si soffre la frammentazione esterna.
Assegnazione concatenata: Ogni file è composto da una lista concatenata di blocchi del disco i quali possono essere sparsi in qualsiasi punto del disco stesso. La directory contiene un puntatore al primo e all'ultimo blocco del file, e ogni blocco contiene un puntatoe al blocco successivo.
Ci sono comunque alcuni svantaggi: ad esempio può essere usata in modo efficiente solo per file ad accesso sequenziale.
Assegnazione indicizzata: Risolve il problema di un efficiente accesso diretto, raggruppando i puntatori in una sola locazione: il blocco indice. Si tratta di un vettore di indirizzi di blocchi del disco, cioè l'*i*-esimo elemento del blocco indice punta all'*i*-esimo blocco del file.

Gestione dello spazio libero e ripristino

Per tenere traccia dello spazio libero in un disco il sistema conserva un elenco dei blocchi liberi. Spesso si realizza come una mappa di bit o vettore di bit, dove se il blocco è libero il bit è 1, altrimenti è 0. E' semplice, anche se è efficiente solo se il vettore è nella memoria centrale.
In caso di crollo del sistema la tabella dei file va generalmente persa. Quindi usiamo il verificatore della coerenza che confronta i dati delle directory con quelli contenuti nei blocchi dei dischi, tentando di correggere ogni incoerenza nel file system.
Importante è fare anche dei *backup* dei dati residenti nei dischi, e quindi il *restore* dei dati dalle copie di riserva in caso di perdita file.

NFS

Nel contesto dell'NFS si considera un insieme di stazioni di lavoro interconnesse come un insieme di calcolatori indipendenti con file system indipendenti: lo scopo è condivisione con client-server.
Il protocollo NFS offre un insieme di RPC per operazioni su file remoti che svolgono operazioni di ricerca, lettura e manipolazione. C'è l'assenza dell'informazione di stato, quindi i dati modificati si devono riscrivere nei dischi del server prima che i risultati siano riportati al client, ed inoltre il server deve scrivere tutti i dati dell'NFS in modo sincrono.
La traduzione dei nomi di percorso si compie suddividendo il percorso in nomi componenti ed eseguendo una chiamata lookup dell'NFS separata per ogni coppia formata da un nome componente e un vnode di directory.

Gestione della memoria

La memoria consiste in un ampio vettore di parole o byte, ciascuno con il proprio indirizzo. La CPU preleva le istruzioni dalla memoria sulla base del contenuto del contatore di programma; tali istruzioni possono determinare ulteriori letture (*load*) e scritture (*store*) in specifici indirizzi della memoria.
L'associazione di istruzioni e dati a indirizzi di memoria si può compiere in qualsiasi fase del seguente percorso: COMPILAZIONE, CARICAMENTO e ESECUZIONE.

Swapping

Per essere eseguito un processo può essere trasferito temporaneamente nella memoria ausiliaria (*backing store*) da cui si riporta nella memoria centrale al momento di riprenderne l'esecuzione.
La memoria ausiliaria deve essere abbastanza ampia da contenere le copie di tutte le immagini di memoria di tutti i processi utenti, e deve permettere un accesso diretto alle immagini di memoria.
Introduciamo quindi il concetto di **roll out**, **roll in** che è una variante del criterio di avvicendamento, basato su algoritmi di scheduling con priorità: un processo con priorità bassa si può scaricare dalla memoria centrale per far spazio ad uno con maggiore priorità che viene eseguito.

Assegnazione contigua della memoria

La memoria centrale di solito si divide in due partizioni, una per il S.O. e una per i processi utenti. Il S.O. di solito si colloca nella memoria bassa, seguendo il vettore delle interruzioni. Quindi i processi utenti sono nella memoria alta.

Protezione della memoria (single partition**)**

La protezione si può realizzare usando un registro di rilocazione, con un registro di limite. Il registro di rilocazione contiene il valore dell'indirizzo fisico maggiore; il registro di limite contiene l'intervallo di indirizzi logici.
Ognuno di quest'ultimi deve essere minore del contenuto del registro di limite.

Assegnazione memoria (multiple partition**)**

Dividiamo la memoria in partizioni di dimensione fissa. Ogni partizione deve contenere esattamente un processo.
Inizialmente tutta la memoria è a disposizione dei processi utenti: si tratta di un buco (***hole***).
Quando si carica un processo occorre cercare un buco sufficientemente grande da contenerlo. Si assegna solo la parte di memoria necessaria a contenerlo, la parte rimanente si usa per soddisfare eventuali richieste future.
E' un'istanza del più generale problema di assegnazione dinamica della memoria, che consiste nel soddisfare una richiesta di dimensione n data una lista di buchi liberi. Le soluzioni più usate sono:

- Si assegna il primo buco abbastanza grande
- Si assegna il buco più piccolo in grado di contenere il processo
- Si assegna il buco più grande.

Frammentazione

Interna: la memoria assegnata può essere leggermente maggiore della memoria richiesta. La memoria è interna a una partizione ma non è in uso.
Esterna: si ha se lo spazio di memoria totale è sufficiente per soddisfare una richiesta, ma non è contiguo. Per risolvere il problema della formattazione esterna si usa la compattazione: riordinare il contenuto della memoria per riunire la memoria libera in un unico grosso blocco.
Possibile solo se la rilocazione è dinamica e si compie nella fase di esecuzione.

Paginazione

Metodo di gestione della memoria che permette che lo spazio degli indirizzi fisici di un processo non sia contiguo. Si suddivide la memoria fisica in blocchi di memoria di dimensioni fisse e la memoria logica in blocchi di eguale dimensione deti pagine. Quando si deve eseguire un processo, si caricano le sue pagine nei blocchi di memoria disponibili, prendendole dalla memoria ausiliaria, che è divisa in blocchi di dimensione fissa, uguale a quella dei blocchi di memoria.
L'indirizzo logico è formato dal numero di pagina e dallo scostamento di pagina.
Con la paginazione si può evitare la frammentazione esterna: qualsiasi blocco di memoria libero si può assegnare ad un processo che ne abbia bisogno; tuttavia si può avere la frammentazione interna.
Un aspetto importante è la distinzione tra la memoria vista dall'utente e l'effettiva memoria fisica: questa differenza è colmata dall'architettura di traduzione degli indirizzi, che fa corrispondere gli indirizzi fisici agli indirizzi logici generati dai processi utenti.

Architettura

L'achitettura d'ausilio alla tabella delle pagine si può realizzare in modi diversi. Si può usare uno specifico insieme di registri, che devono poter operare a una velocità molto elevata per un'efficiente traduzione degli indirizzi.
Non si possono impiegare registri veloci per realizzare la tabella delle pagine; quest'ultima si mantiene nella memoria principale e un registro di base della tabella delle pagine (PTBR) punta alla tabella stessa; il cambio delle tabelle richiede soltanto di modificare questo registro. Però questo presenta un problema connesso al tempo necessario per accedere a una locazione della memoria utente e la soluzione consiste nell'impiego di una piccola cache di ricerca veloce, detta TLB.
Consiste di due parti: una chiave e un valore: quando si presenta un elemento lo si confronta con tutte le chiavi e se trova una corrispondenza, riporta il valore correlato.

Protezione

La protezione della memoria è assicurata dai bit di protezione associati a ogni blocco di memoria, che si trovano normalmente nella tabella delle pagine.
Di solito si associa un ulteriore bit, detto di validità, a ciascun elemento della tabella delle pagine. Tale bit impostato:

- Valido: indica che la pagina corrispondente è nello spazio d'indirizzi logici del processo, quindi è una pagina valida.
- Non valido: indica che la pagina non è nello spazio d'indirizzi logici del processo.

Quindi si possono riconoscere indirizzi illegali e notificarne la presenza attraverso un segnale d'eccezione.
Alcune architetture dispongono di registri, detti registri di lunghezza della tabella delle pagine (PTLR), per indicarne la dimensione. Questo valore serve per verificare che un indirizzo logico si trovi nell'intevallo valido per il processo.

Segmentazione

E' uno schema di gestione della memoria che consente di gestire una visione della memoria da parte dell'utente fatta di segmenti di programma di lunghezza variabile.
Uno spazio di indirizzi logici è una raccolta di segmenti, ciascuno dei quali ha un nome e una lunghezza. Gli indirizzi logici specificano sia il nome sia lo scostamento all'interno del segmento, quindi ho una coppia ordinata di valori, quindi ogni riferimento si compie per mezzo di un numero anzichè di un nome.

Architettura

La memoria fisica è una sequenza di byte unidimensionale e quindi bisogna tradurre gli indirizzi bidimensionali definiti dall'utente. Lo facciamo tramite la tabella dei segmenti: ogni suo elemento è una coppia ordinata dove ho la **base** del segmento, contenente l'indirizzo fisico iniziale della memoria nel quale il segmento risiede (STBR), e il **limite** del segmento, che contiene la lunghezza del segmento (STLR).

Protezione e condivisione

Un vantaggio particolare dato dalla segmentazione è l'associazione della protezione con i segmenti, in quanto è probabile che tutti gli elementi del segmento siano usati nello stesso modo. L'architettura di traduzione degli indirizzi della memoria controlla i bit di protezione associati a ogni elemento della tabella dei segmenti, in modo da impedire accessi illegali alla memoria.ù
Per quanto riguarda la condivisione ogni processo ha una tabella di segmenti che si condividono se gli elementi delle tabelle dei segmenti di due processi diversi puntano alle stesse locazioni fisiche.
Si può condividere qualsiasi informazione definita come segmento e quindi anche un programma composto da più segmenti.

Frammentazione

La segmentazione può causare anche frammentazione esterna se tutti i blocchi di memoria libera sono troppo piccoli perchè contengano un segmento. In questo caso è possibile ritardare il processo fino a che non si renda disponibile più memoria, oppure finchè la compattazione crei un buco più grande.
Se lo scheduler della CPU deve attendere un processo a causa di un problema di assegnazione di memoria, può esaminare la coda d'attesa per la CPU alla ricerca di un processo da eseguire che sia più piccolo e con priorità più bassa.

Memoria virtuale

Permette che i programmi siano più grandi della memoria fisica. Inoltre separa la memoria logica dalla memoria fisica.

Paginazione su richiesta

Il paginatore, che è il modulo del S.O. che si occupa della sostituzione delle pagine, trasferisce nella memoria solo le pagine che ritiene necessarie, evitando quelle che non sono effettivamente usate.
Bisogna distinguere le pagine presenti nella memoria da quelle nei dischi, e si può usare lo schema del bit di validità, dove con il bit impostato come "non valido" indichiamo che la pagina non appartiene allo spazio di indirizzi logici del processo, oppure è nel disco.
L'accesso a una pagina contrassegnata non valida causa un'eccezione di pagina mancante (*page fault trap*); tale eccezione è dovuta a un insuccesso nella scelta delle pagine da caricare, detto assenza di pagina (*page fault*), cui bisogna rimediare.
Bisogna salvare lo stato del processo: il S.O. carica nella memoria la pagina desiderata e riavvia il processo come se tale pagina fosse già presente nella memoria.
I meccanismi d'ausilio sono:

- **TABELLA DELLE PAGINE**: capacità di contrassegnare un bit come non valido o un valore speciale dei bit di protezione.
- **MEMORIA AUSILIARIA**: conserva le pagine che non sono presenti nella memoria centrale.

Il sistema di paginazione si colloca tra la CPU e la memoria di un calcolatore e deve essere completamente trasparente al processo utente. Inoltre consente l'esecuzione di processi i cui requisiti di spazio di memoria superano la memoria fisica disponibile (si eseguono nella memoria virtuale).

Sostituzione delle pagine

Segue il seguente criterio: se nessun blocco di memoria è libero, si può liberarne uno attualmente inutilizzato, scrivendo il suo contenuto nell'area d'avvicendamento e modificando la tabella delle pagine.
Se non esiste alcun blocco di memoria libero, sono necessari due trasferimenti di pagine. Questo sovraccarico si può ridurre usando un bit di modifica (*dirty bit*).
Esistono molti algoritmi di sostituzione delle pagine: generalmente si sceglie quello con la minima frequenza delle assenze di pagine su una particolare successione di riferimenti alla memoria.

Sostituzione secondo ordine d'arrivo

E' un algoritmo FIFO, che associa a ogni pagina l'istante di tempo in cui quella pagina è stata portata nella memoria. Se si deve sostituirne una, si seleziona quella presente nella memoria da più tempo. Con alcuni algoritmi di sostituzione delle pagine la frequenza delle assenze delle pagine può aumentare con l'aumentare del numero di blocchi di memoria assegnati: anomalia di Belady.

Sostituzione delle pagine ottimale

Non presenta mai l'anomalia di Belady, in quanto assicura la frequenza di assenze di pagine più bassa possibile per un numero fissato di blocchi di memoria. Difficile da realizzare perchè richiede la conoscenza futura della successione dei riferimenti.

Sostituzione pagine usate meno frequentemente

E' un'approssimazione del precedente. Detto algoritmo LRU (*last recently used*) associa a ogni pagina l'istante in cui è stata usata per l'ultima volta, e sostituisce la pagina che non è stata usata per il periodo più lungo. Esistono due implementazioni:

- **Contatori**: alla CPU si aggiunge un contatore che si incrementa a ogni riferimento alla memoria. Si sostituisce la pagina con il valore associato più piccolo.
- **Pila**: ogni volta che si fa riferimento a una pagina, la si estrae dalla pila e la si colloca in cima a quest'ultima.

Sostituzione pagine per approssimazione a LRU

1. Si può usare un bit di riferimento. Inizialmente sono tutti impostati a 0. Viene impostato a 1 il bit associato a ciascuna pagina cui si fa riferimento; quindi si può sapere le pagine usate, ma non il loro ordine
2. Seconda chance: dopo la selezione si controlla il bit; se è 0 si sostituisce la pagina, se è 1 la selezione passa alla successiva pagina FIFO. Quando una pagina riceve la seconda chance, si azzera il suo bit di riferimento e si aggiorna il suo istante d'arrivo al momento attuale.

Sostituzione pagine basata su conteggio

Si usa un contatore del numero di riferimenti che sono stati fatti a ciascuna pagina:

- LFU: richiede che si sostituisca la pagina con il conteggio più basso, visto che una pagina uscita attivamente deve avere un conteggio di riferimento alto.
- MFU: basato sul fatto che la pagina col contatore più basso è stata appena inserita e non uscita ancora.

Assegnazione dei blocchi di memoria

C'è una strategia di base: al processo utente si assegna qualsiasi blocco di memoria libero.
Le strategie di assegnazione sono soggette a parecchi vincoli, fra i quali c'è quello di assegnare almeno un numero minimo di blocchi di memoria.
I blocchi di memoria disponibili devono essere in numero sufficiente per contenere tutte le pagine cui ogni singola istruzione può fare riferimento.
Occorre un criterio di assegnazione dei blocchi di memoria, e per questo ci sono due categorie generali. La sostituzione globale permette che per un processo si scelga un blocco di memoria per la sostituzione dall'insieme di tutti i blocchi di memoria (è un metodo di assegnazione statica); la sostituzione locale richiede invece che per ogni processo si scelga un blocco di memoria solo dal suo insieme di blocchi di memoria (è un metodo di assegnazione dinamica).

Attività di paginazione degenere

Il modello dell'insieme di lavoro presuppone che i processi siano eseguiti in località. L'insieme di lavoro è l'insieme delle pagine nella località corrente. A ogni processo si possono assegnare blocchi di memoria sufficienti al suo corrente insieme di lavoro. Se un processo non ha spazio di memoria sufficiente per il prorpio insieme di lavoro, si ha una degenerazione dell'attività di paginazione (*thrashing*). Se a ogni processo si devono fornire blocchi di memoria sufficienti per evitare tale degenerazione, sono necessarie le attività d'avvicendamento (*swapping*) e scheduling dei processi.