

Interfata medii stocare

put(address, data)

get(address)



Sisteme de Fișiere
Organizarea datelor pe un suport fizic
în funcție de caracteristicile suportului
- Fișiere, Directoare, arbori
- Metode de accesare a datelor
- Metode de organizare a datelor

Sisteme de Fișiere

SO Curs 12

Obiective de învățare

- descrierea sistemelor de fișiere
- descrierea fișierelor
- descrierea arborilor
- descrierea metodelor de accesare a datelor



Alocarea spațiului pe disc
Cum sunt organizate datele pe disc?
Alocarea spațiului pe disc
- Metode de accesare a datelor
- Metode de organizare a datelor

Implementarea direcțiilor
- Metode de accesare a datelor
- Metode de organizare a datelor
- Metode de accesare a datelor

Gestiunea spațiului liber
Cum funcționează?
- Metode de accesare a datelor
- Metode de organizare a datelor

Considerente de Performanță
- Metode de accesare a datelor
- Metode de organizare a datelor

Convenții utilizate în sistemele de fișiere
- Metode de accesare a datelor
- Metode de organizare a datelor

Concluzii
- Metode de accesare a datelor
- Metode de organizare a datelor

Trei abstractii fundamentale

Memorie virtuala

Iluzia unei memorii continue

Procese

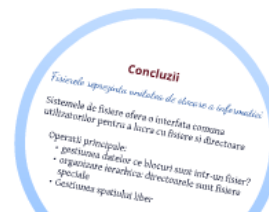
Unitate de executie

Sisteme de Fisiere

SO Curs 12

Sistem de fisiere
Organizarea datelor pe un suport persistent
Oferea utilizatorilor, facilități de administrare:
- fișiere, directori, servicii
- atribuiri (drepturi, permisiuni, drept acces)
- servicii de securitate: controlul disponibilității
- fișierelor

Discuție cu privire la TEHET
Discuție cu privire la utilizarea de resurse și servicii:
- fișiere
- servicii
- servicii
- servicii
- servicii
- servicii



Fisiere

Orice este un fisier in UNIX

Fisierul este unitatea de stocare a informatiei:

- date
- metadate

Ofera:

- persistenta datelor
- partajarea datelor
- protectia datelor

Sistem de fisiere

Organizeaza datele pe un suport persistent

Ofera o interfata facila utilizatorului:

- fisiere, directoare, ierarhie
- attribute (drepturi, nume, timpi acces)

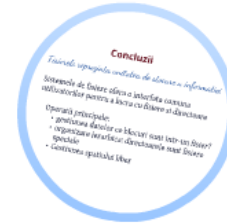
Ascunde utilizatorilor detaliile dispozitivelor de stocare

Sisteme de Fișiere
Organizarea datelor pe un suport fizic
Algoritmii de accesare a datelor
- Metode de accesare a datelor
- Metode de accesare a datelor
- Metode de accesare a datelor

Sisteme de Fișiere

SO Curs 12

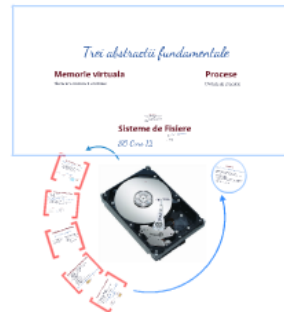
Obiective de învățare
- descrierea
- descrierea
- descrierea



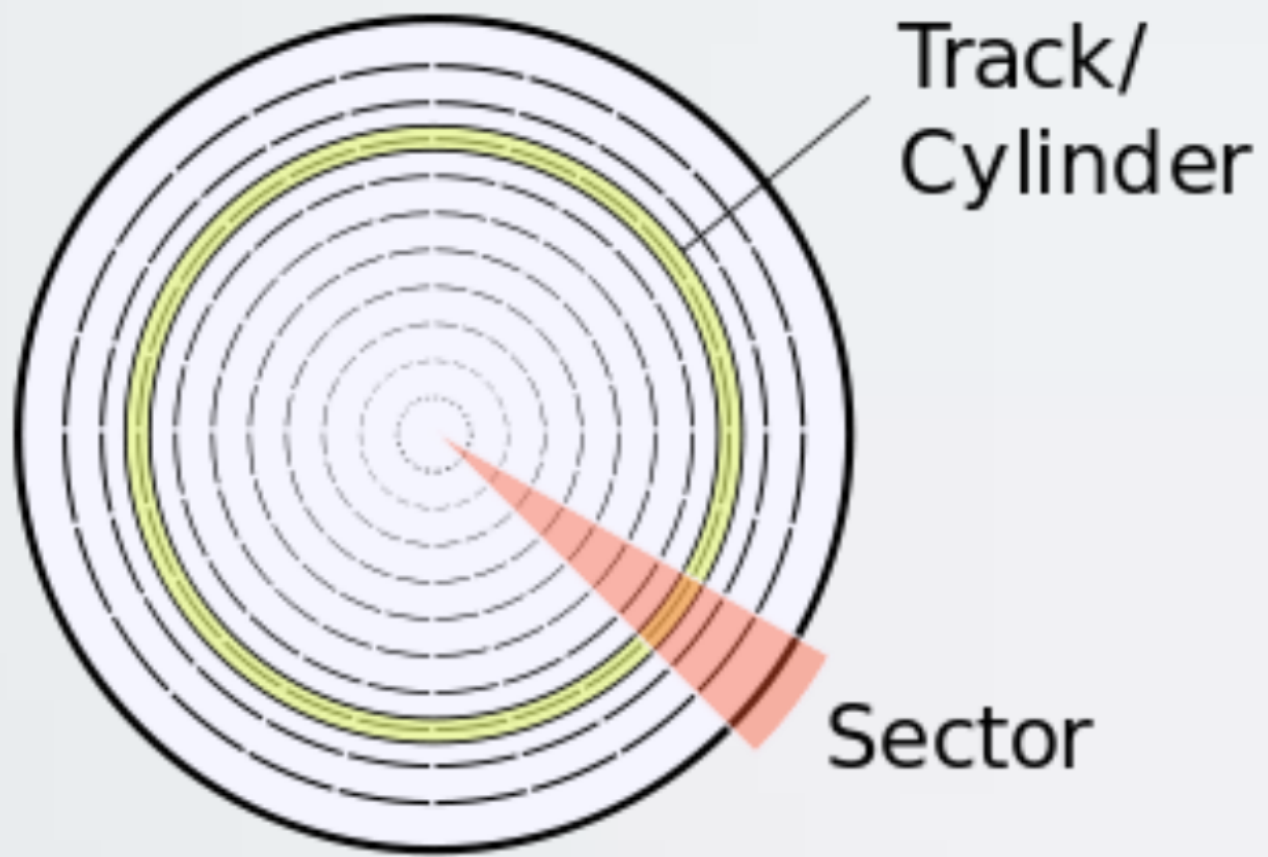
Interfata medii stocare

put(address, data)

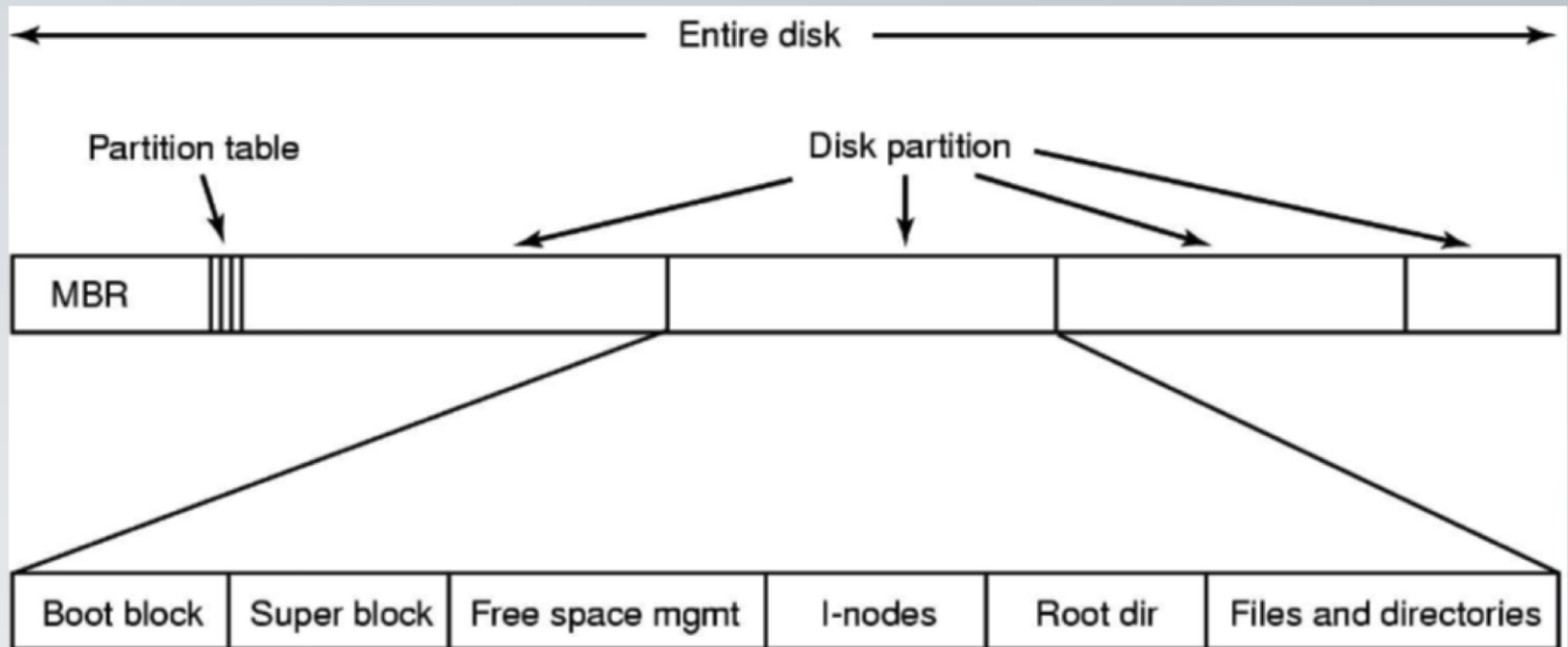
get(address)







Organizarea unui Hard Disc Drive



Alocarea spatiului pe disc

Cum tinem minte unde sunt datele fiecarui fisier?

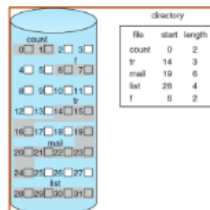
Alocarea se face la nivel de bloc, care este:

- multiplu de sector (512B)
- submultiplu de pagina (4096B)

Solutii posibile

Alocare Contigua

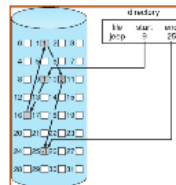
Fisierele stocate
continuu pe disc



Avantaje?
Dezavantaje?

Alocare cu liste

Un bloc contine pointer
catre urmatorul bloc

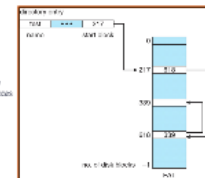


Avantaje?
Dezavantaje?

File Allocation Table

O zona separata contine
pointeri catre blocurile
fisierei!

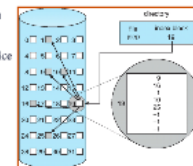
- Cost mare de memorie
- Se reduce timpul de acces



Alocare indexata

Un index block pentru
fiecare fisier (index)

Mapeaza blocurile logice
in blocuri fizice



Avantaje?
Dezavantaje?

Alocare Contigua

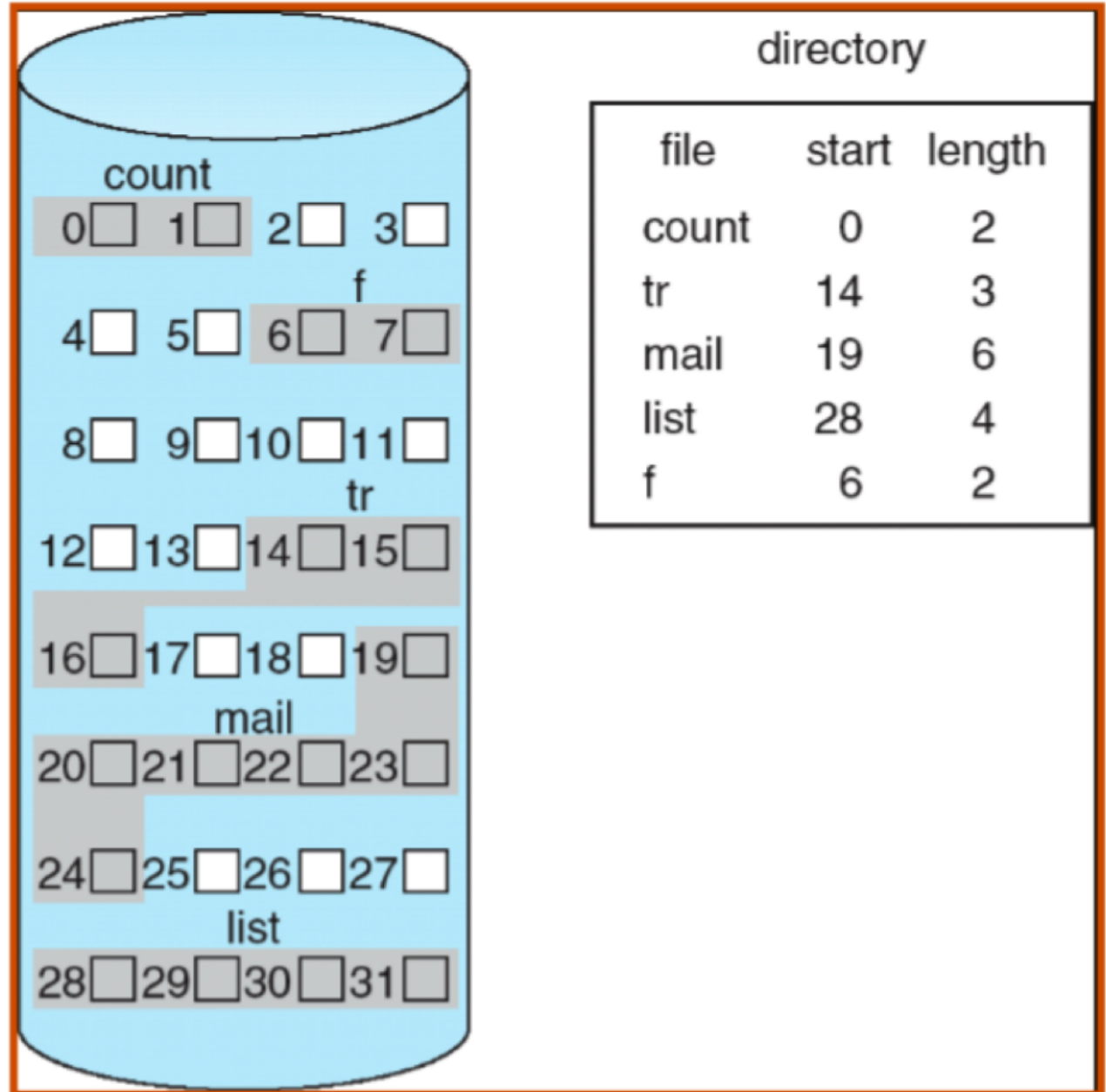
Fisierele stocate
continuu pe disc

Avantaje?

- Identificare simpla a blocurilor
- Viteza mare la citire

Dezavantaje?

- Fragmentare externa
- Trebuie specificata dimensiunea fisierului la creare



Avantaje?

- Identificare simpla a blocurilor
- Viteza mare la citire

- Identificare simpla a blocurilor
- Viteza mare la citire

Dezavantaje?

- Fragmentare externa
- Trebuie specificata dimensiunea fisierului la creare

Alocare cu liste

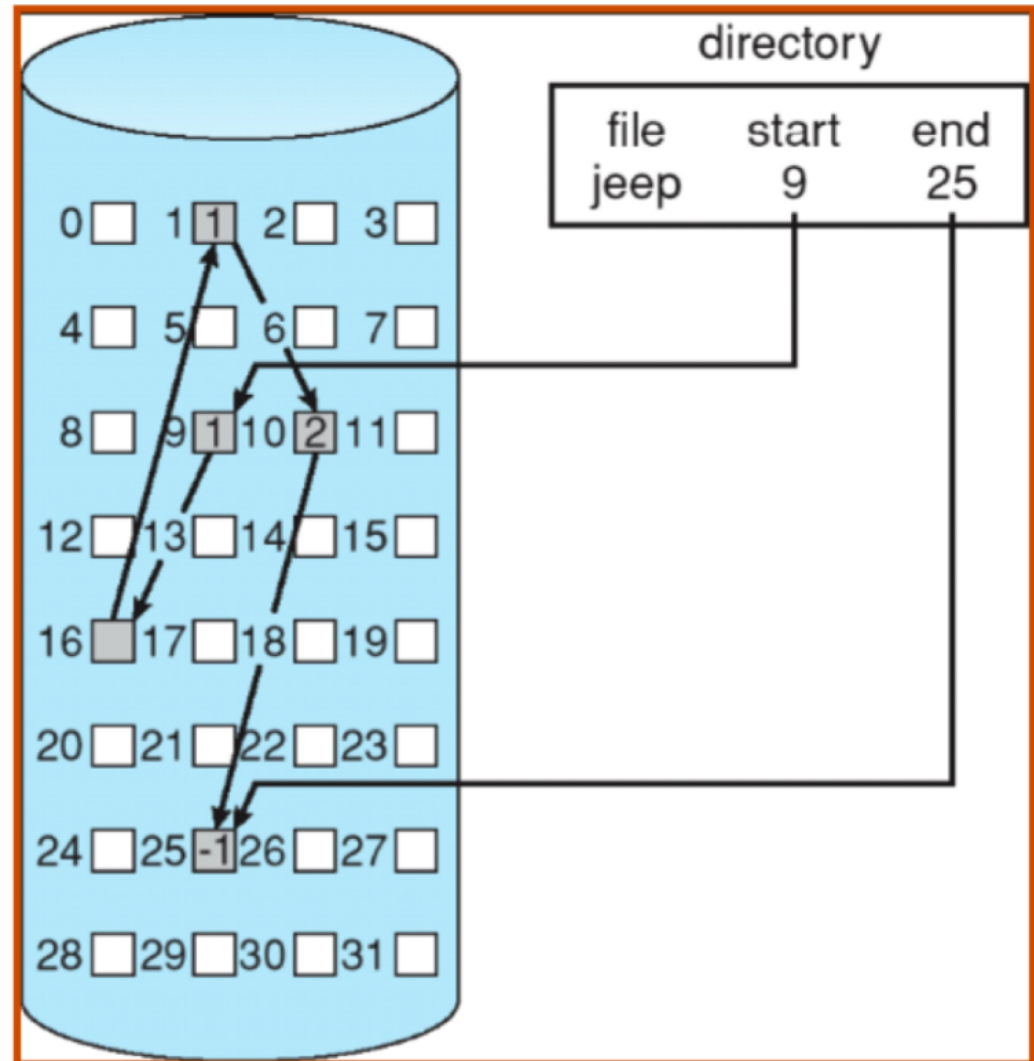
Un bloc contine pointer
catre urmatorul bloc

Avantaje?

- nu mai exista fragmentare externa
- este necesar doar primul bloc pentru a localiza fisierul

Dezavantaje?

- Timp de access ridicat pentru ultimele blocuri
- Performanta scazuta la acces aleator



Avantaje?

- nu mai exista fragmentare externa
- este necesar doar primul bloc pentru a localiza fisierul

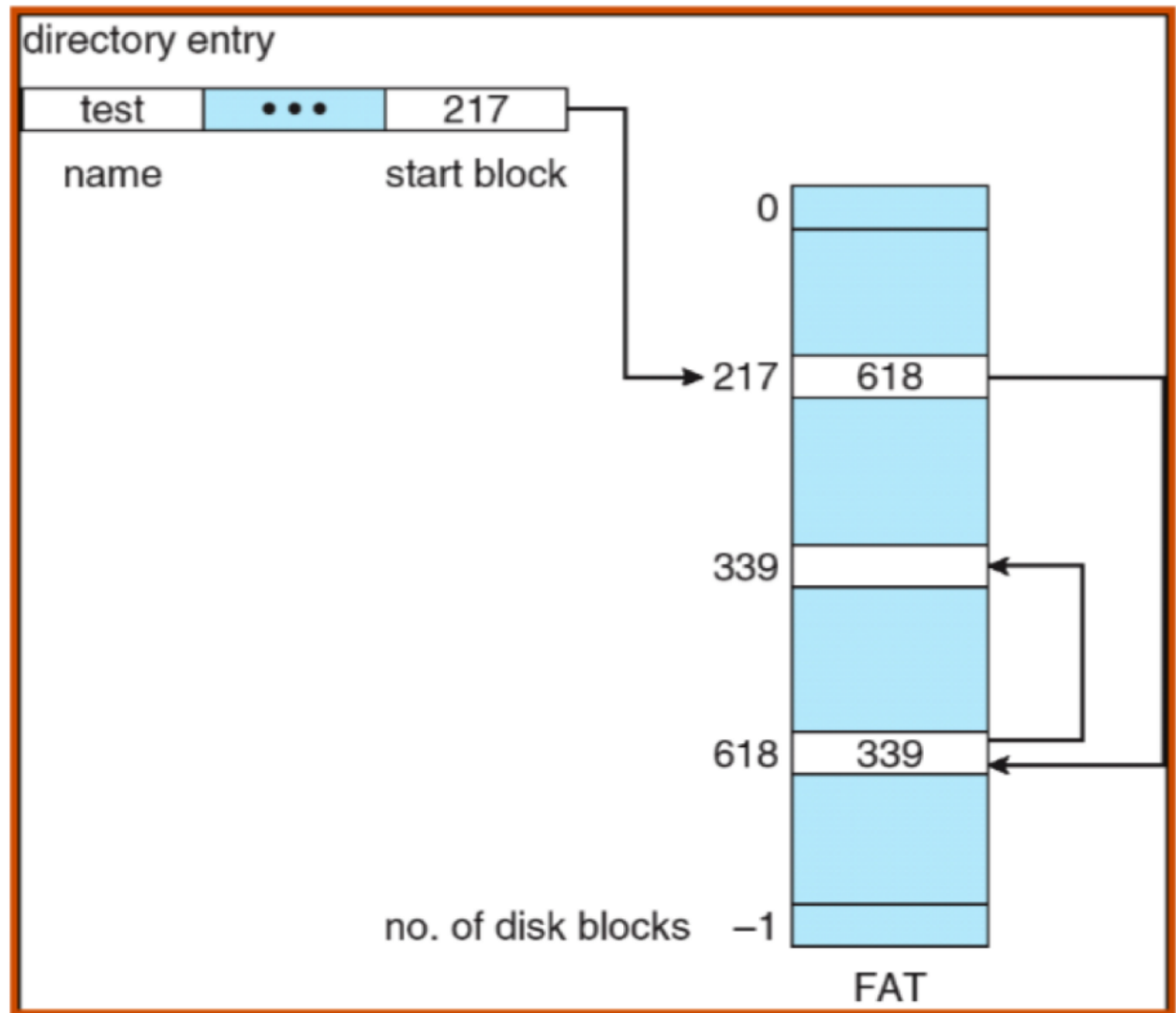
Dezavantaje?

- Timp de access ridicat pentru ultimele blocuri
- Performanta scazuta la acces aleator

File Allocation Table

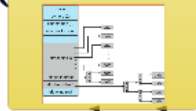
O zona separata contine pointerii catre blocurile fisierului

- Cost mare de memorie
- Se reduce timpul de acces



Alocare indexata

Un index block pentru fiecare fisier (**i-node**)



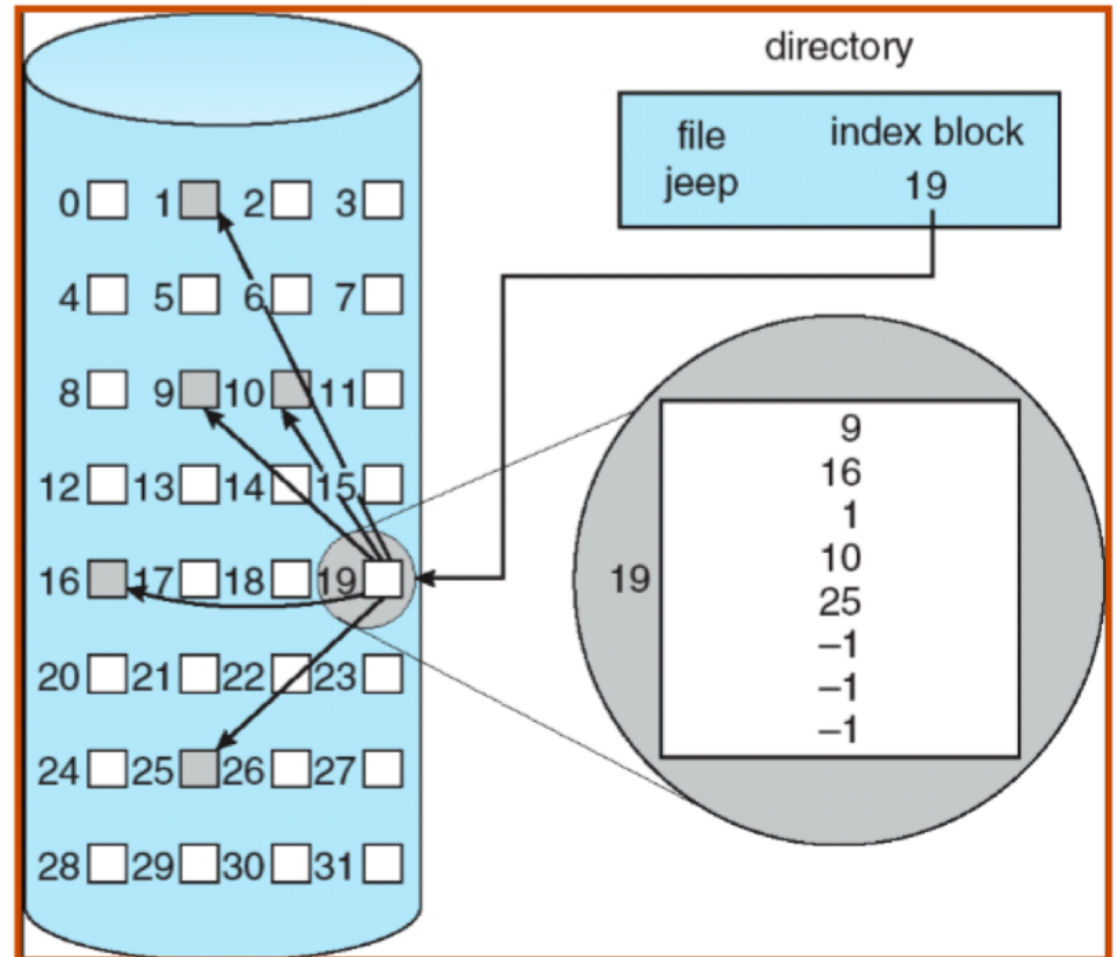
Mapeaza blocurile logice in blocuri fizice

Avantaje?

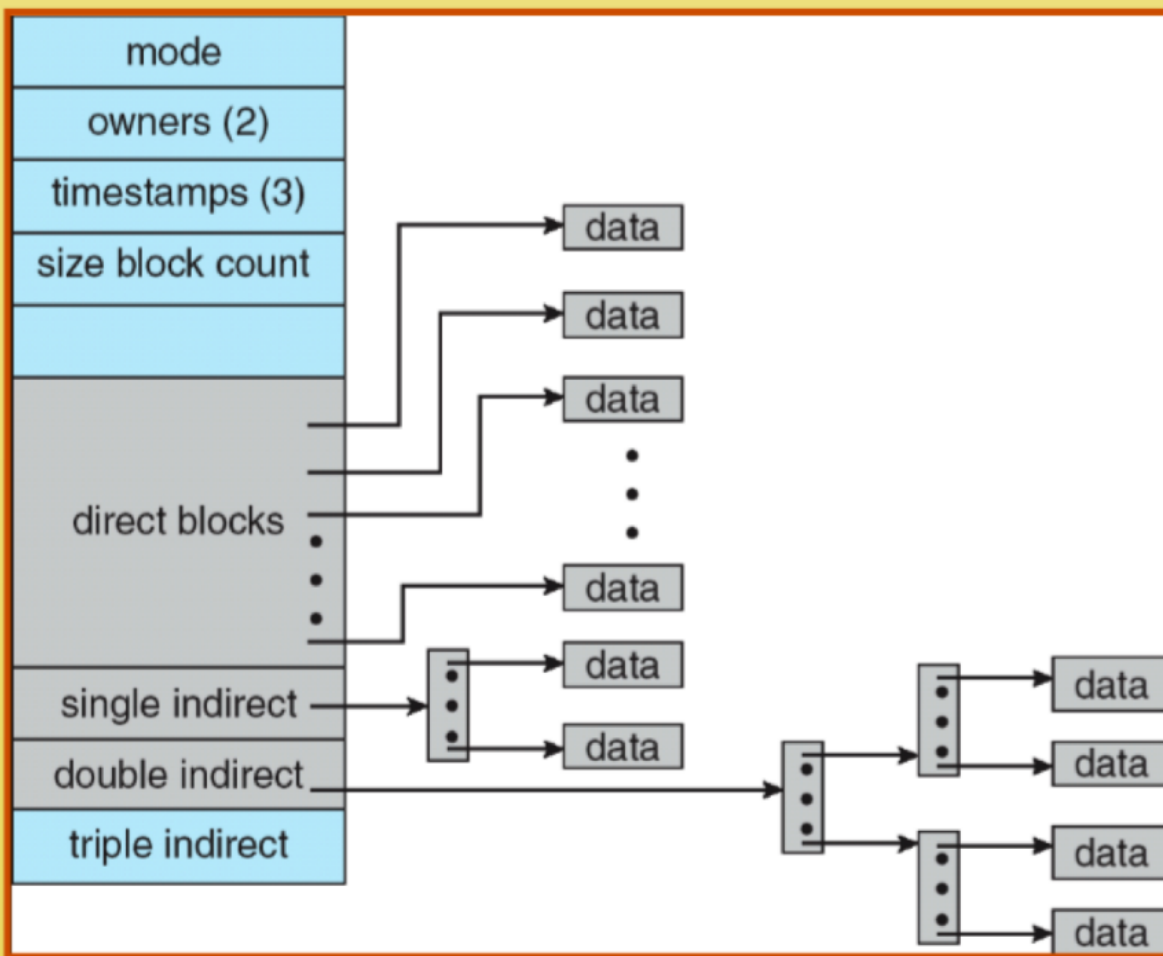
- nu exista fragmentare externa
- timp de acces bun

Dezavantaje?

- numarul limitat de intrari in tabela de indexare
- ultimele intrari contin pointeri catre alte tabele de indici (creste latenta)

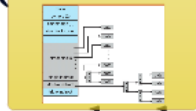


i-node



Alocare indexata

Un index block pentru fiecare fisier (**i-node**)



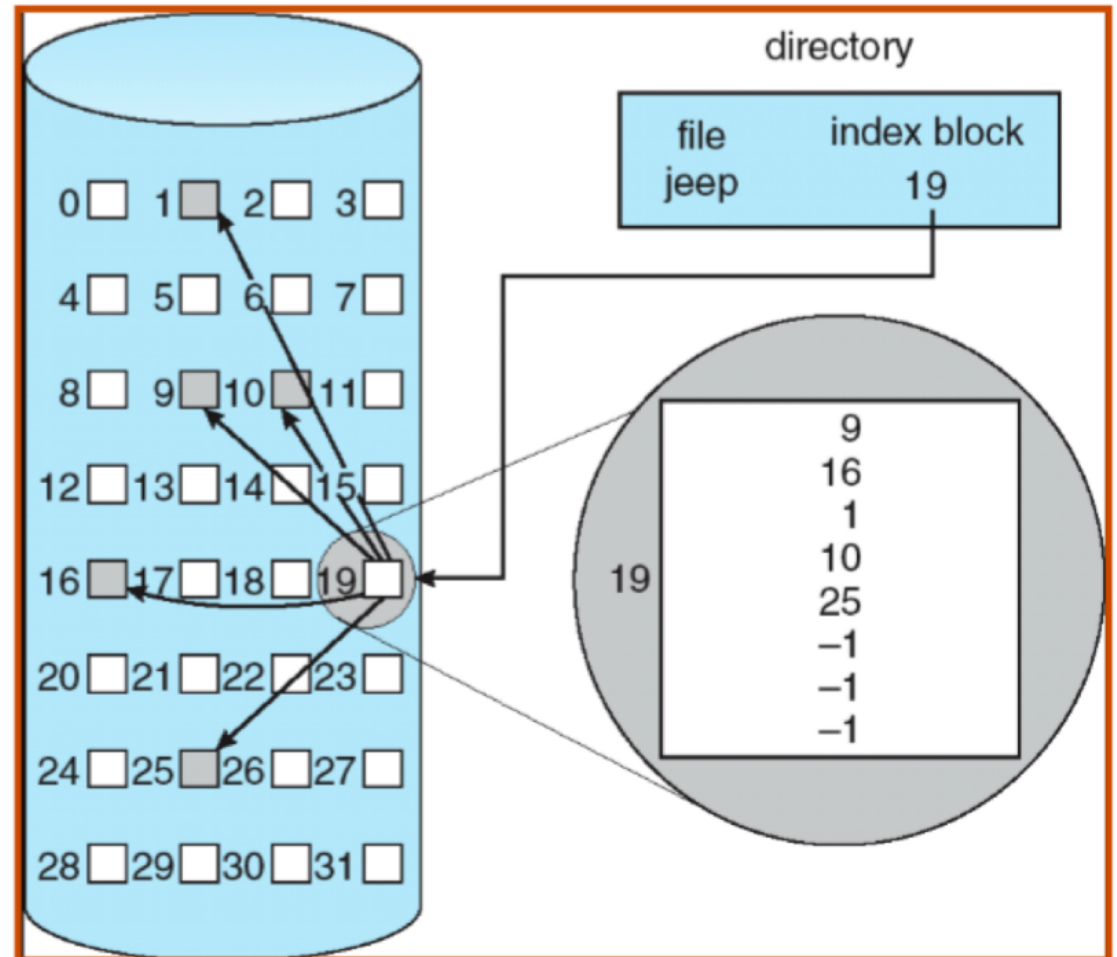
Mapeaza blocurile logice in blocuri fizice

Avantaje?

- nu exista fragmentare externa
- timp de acces bun

Dezavantaje?

- numarul limitat de intrari in tabela de indexare
- ultimele intrari contin pointeri catre alte tabele de indici (creste latenta)



Avantaje?

- nu exista fragmentare externa
- timp de acces bun

Dezavantaje?

- numarul limitat de intrari in tabela de indexare
- ultimele intrari contin pointeri catre alte tabele de indecsi (creste latentia)

Alocarea spatiului pe disc

Cum tinem minte unde sunt datele fiecarui fisier?

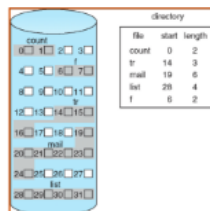
Alocarea se face la nivel de bloc, care este:

- multiplu de sector (512B)
- submultiplu de pagina (4096B)

Solutii posibile

Alocare Contigua

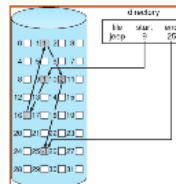
Fisierele stocate
continuu pe disc



Avantaje?
Dezavantaje?

Alocare cu liste

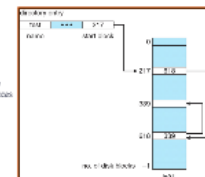
Un bloc contine pointer
 catre urmatul bloc



Avantaje?
Dezavantaje?

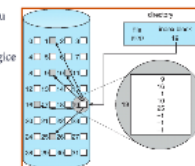
File Allocation Table

O zona separata contine
 pointeri catre blocurile
 fisierului



Alocare indexata

Un index block pentru
 fiecare fisier
 Mapeaza blocurile logice
 in blocuri fizice



Avantaje?
Dezavantaje?

Implementarea directoarelor

O structura dentry descrie un fisier:

- nume fisier
- un identificator al file-control block (i-node)

Directorul este un fisier special care contine informatii despre alte fisiere

- Directorul are deasemenea un FCB (i-node, intrare in FAT)
- Intrarile . si .. sunt speciale
- Dimensiunea este data de structurile continute

Exemplu: ext2

	inode	rec_len	file_type	name_len	name
0	21	12	1	2	· \0 \0 \0
12	22	12	2	2	· \0 \0
24	53	16	5	2	h o m e 1 \0 \0 \0
40	67	28	3	2	u s r \0
52	0	16	7	1	o 1 d f i 1 e \0
68	34	12	4	2	s b i n

Link-uri

Permit utilizatorilor sa refere acelasi fisier din mai multe directoare/cu mai multe nume

Hard-links

Doua structuri dentry care refera acelasi i-node
Pot fi folosite intre sisteme de fisiere diferite?

Soft-links

i-node specializat, contine numele fisierului referit

Exemplu: ext2

	inode	rec_len	file_type	name_len	name								
0	21	12	1	2	.	\0	\0	\0					
12	22	12	2	2	.	.	\0	\0					
24	53	16	5	2	h	o	m	e	1	\0	\0	\0	
40	67	28	3	2	u	s	r	\0					
52	0	16	7	1	o	l	d	f	i	l	e	\0	
68	34	12	4	2	s	b	i	n					

Link-uri

Permit utilizatorilor sa refere acelasi fisier din mai multe directoare/cu mai multe nume

Hard-links

Doua structuri dentry care refera acelasi i-node
Pot fi folosite intre sisteme de fisiere diferite?

Soft-links

i-node specializat, contine numele fisierului referit

Gestiunea spatiului liber

Care blocuri sunt libere?

Vector de biti

1 bit pentru fiecare bloc
overhead constant

Liste inlantuite

Primul bloc liber are pointer catre al
doilea

Capatul listei tinut in memorie

Optimizari:

- tabele de blocuri libere
- zone contigue

Considerente de Performanta

Discul este mult mai lent decat memoria:

- 4Gb/s interfata SATA (2Gb/s in practica)
- 100Gb/s memoria RAM

Timp de acces:

- 10ms (disc)
- 1ns (memorie)

Aplicatiile pot citi:

- secvential sau aleator
- mai multe sau mai putine date

Cum imbunatatim performantele sistemelor de fisiere?

Tehnici folosite

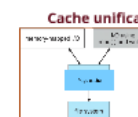
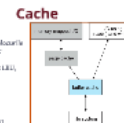
- cache** Evita accesul la disc
- pre-fetch** Optimizare acces secvential
- grupeaza cereri** Reducere timp de seek

Buffer cache

- Memorie in memoria fizica
- reduce viteza de acces
- reduce timpul de wait
- algoritmi de algoritmi LRU, FIFO

Page cache

- Folosirea paginilor in loc de blocuri
- Memory mapped I/O



Tehnici folosite

cache Evita accesul la disc
pre-fetch Optimizare acces secvential
grupeaza Reducere timp de seek
cereri

Ordonarea cererilor

- pentru a creste throughput-ul
- a reduce timpul de seek
- a minimiza inechitatea



Sisteme de Fisiere Log-Structured

- Scrie totul secvential
- Tine minte unde este cea mai recenta copie pentru fiecare bloc
- In paralel, sterge informatiile inutile

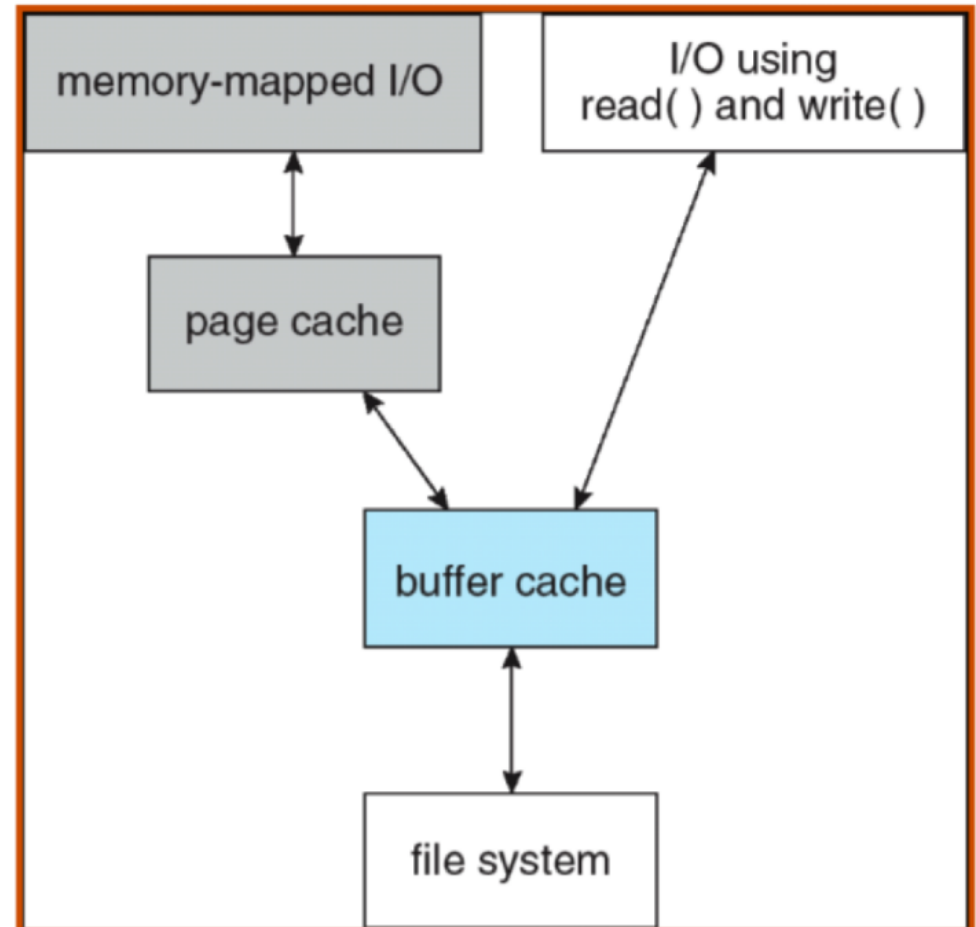
Cache

Buffer cache

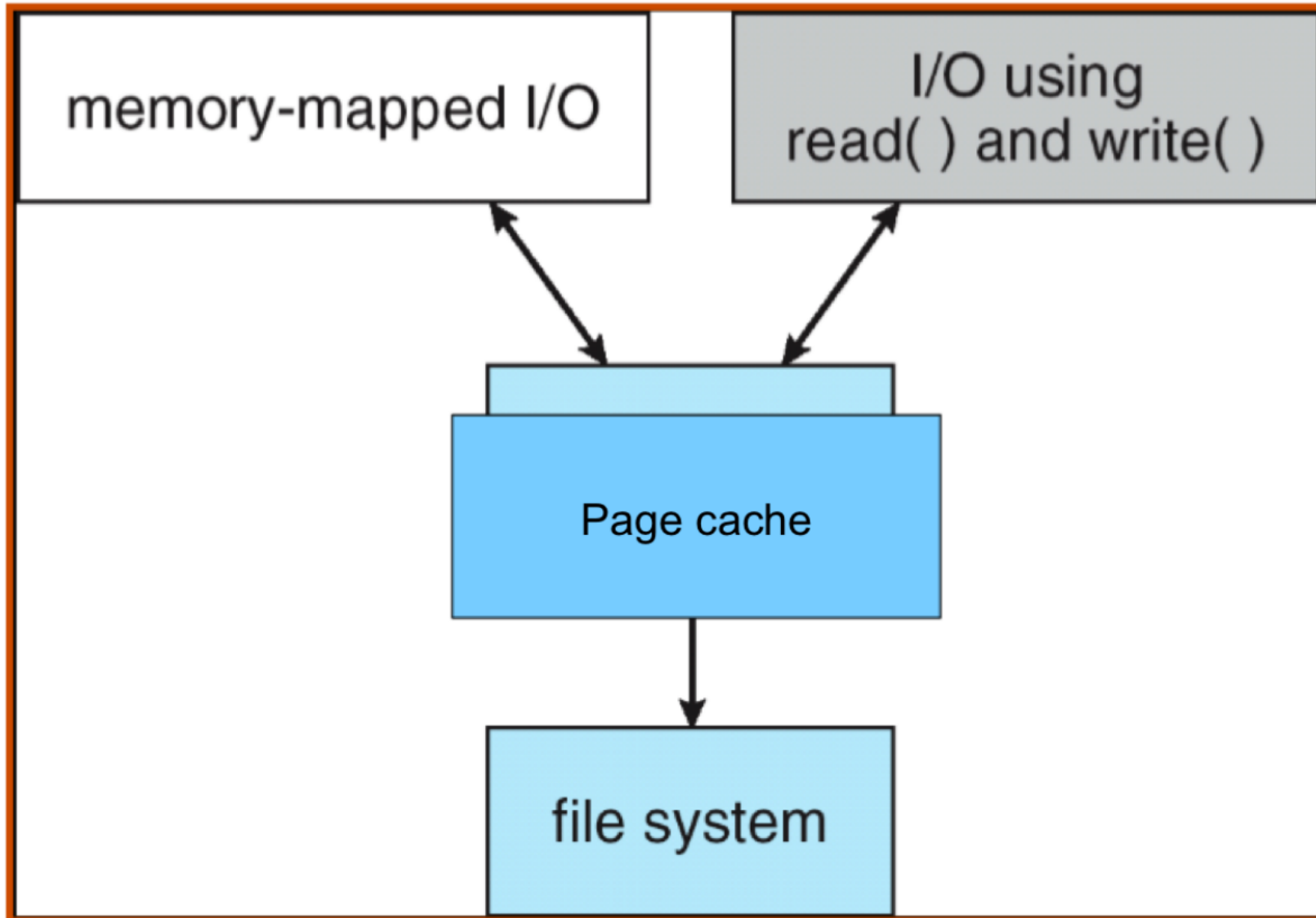
- Mentine in memorie blocurile recent citite de pe disc
- foloseste tabele Hash
- algoritmi de inlocuire: LRU, FIFO

Page cache

- Foloseste pagini in loc de blocuri
- Memory-mapped I/O



Cache unificat



Tehnici folosite

cache Evita accesul la disc
pre-fetch Optimizare acces secvential
grupeaza Reducere timp de seek
cereri

Ordonarea cererilor

- pentru a creste throughput-ul
- a reduce timpul de seek
- a minimiza inechitatea

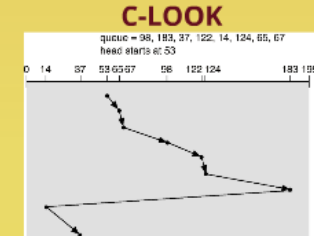
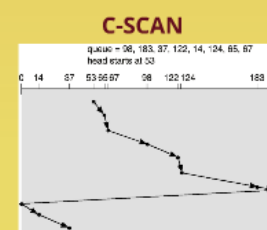
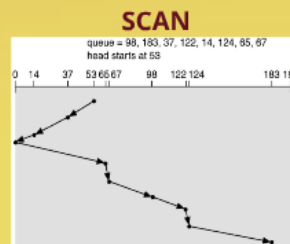
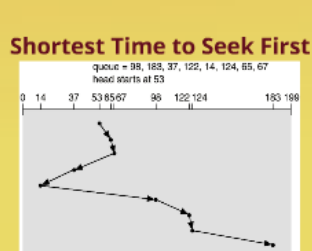
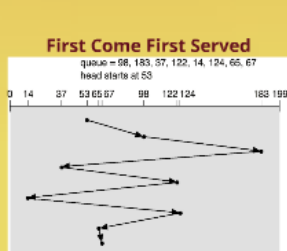


Sisteme de Fisiere Log-Structured

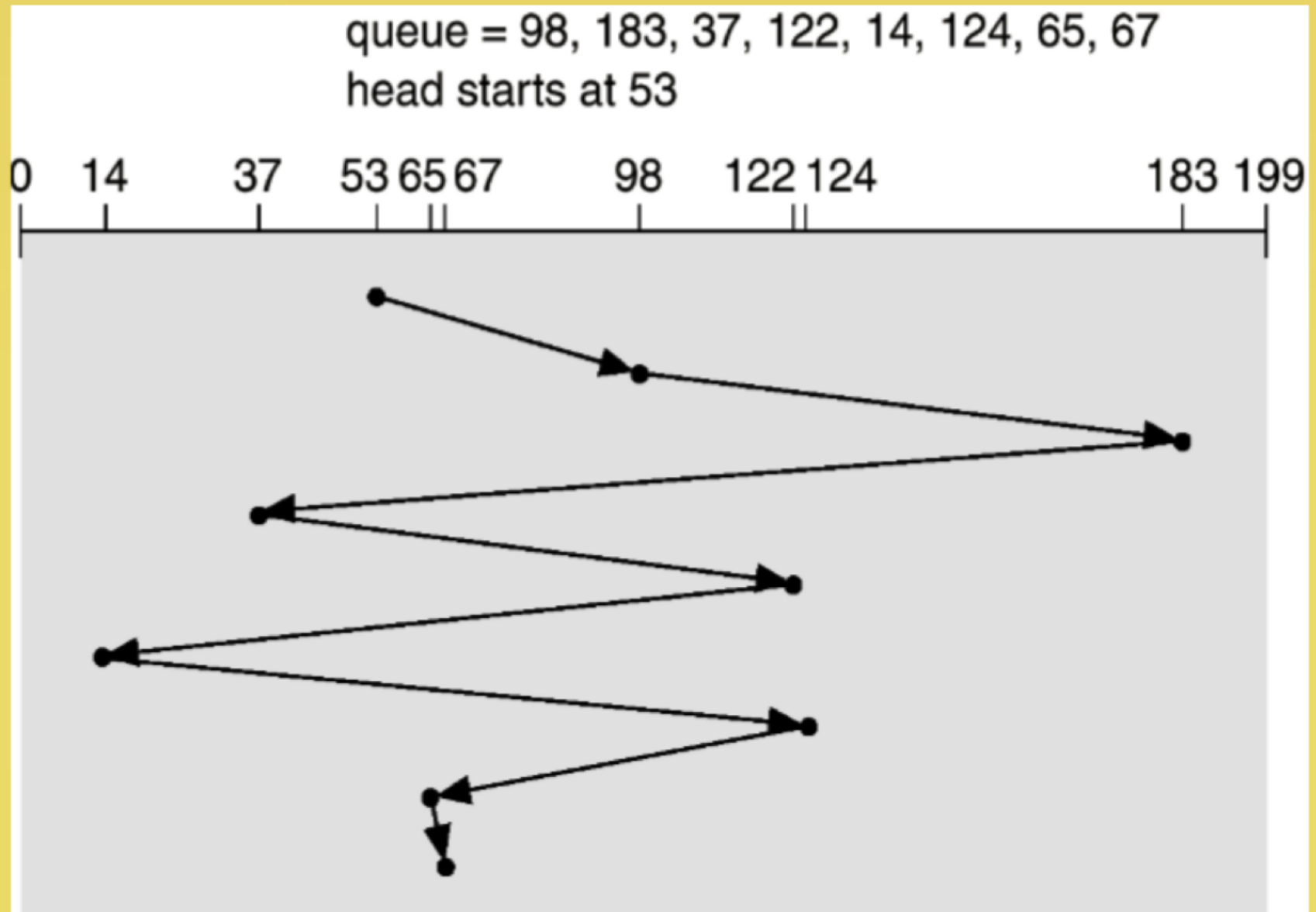
- Scrie totul secvential
- Tine minte unde este cea mai recenta copie pentru fiecare bloc
- In paralel, sterge informatiile inutile

Ordonarea cererilor

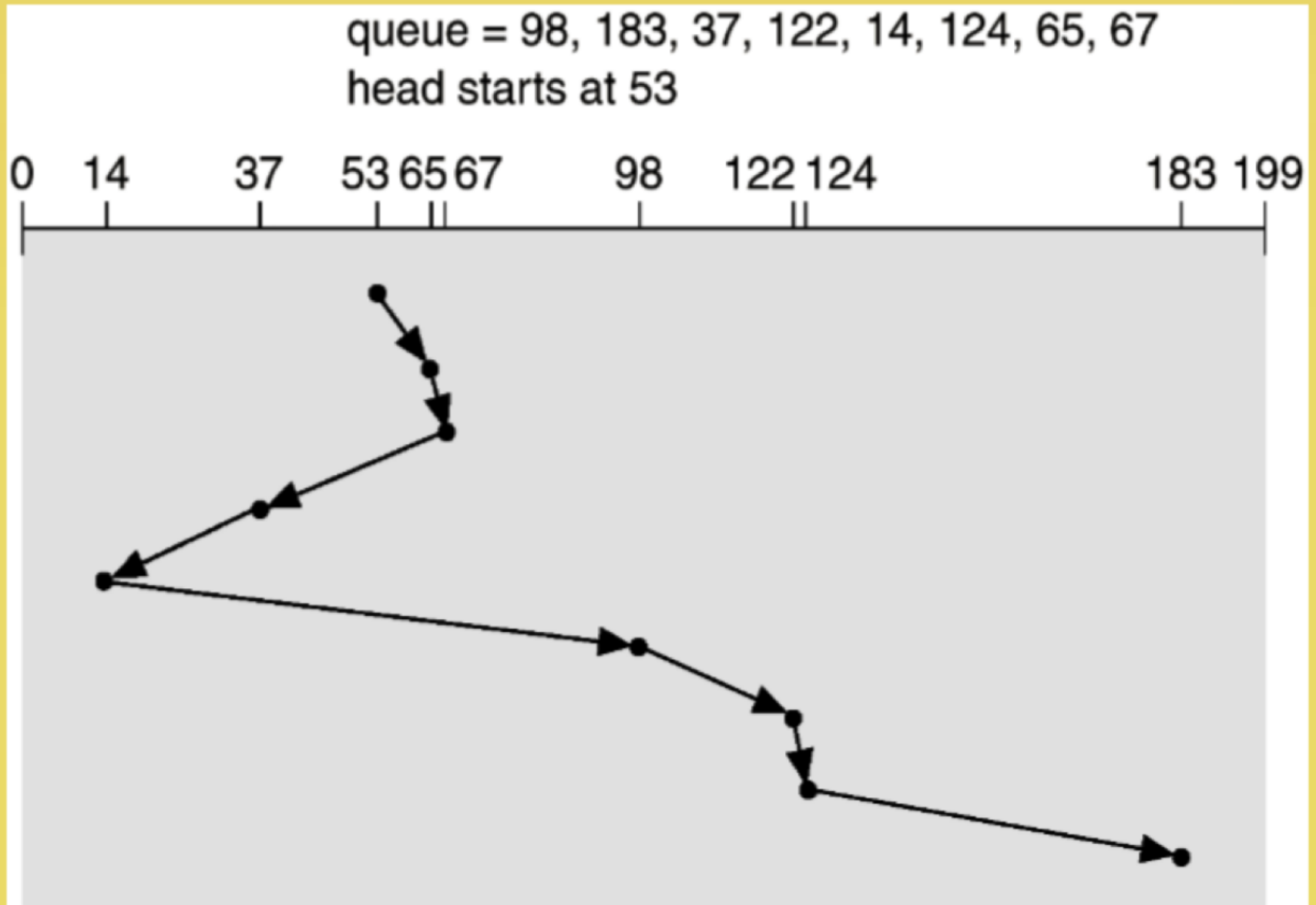
- pentru a crește throughput-ul
- a reduce timpul de seek
- a minimiza inechitatea



First Come First Served

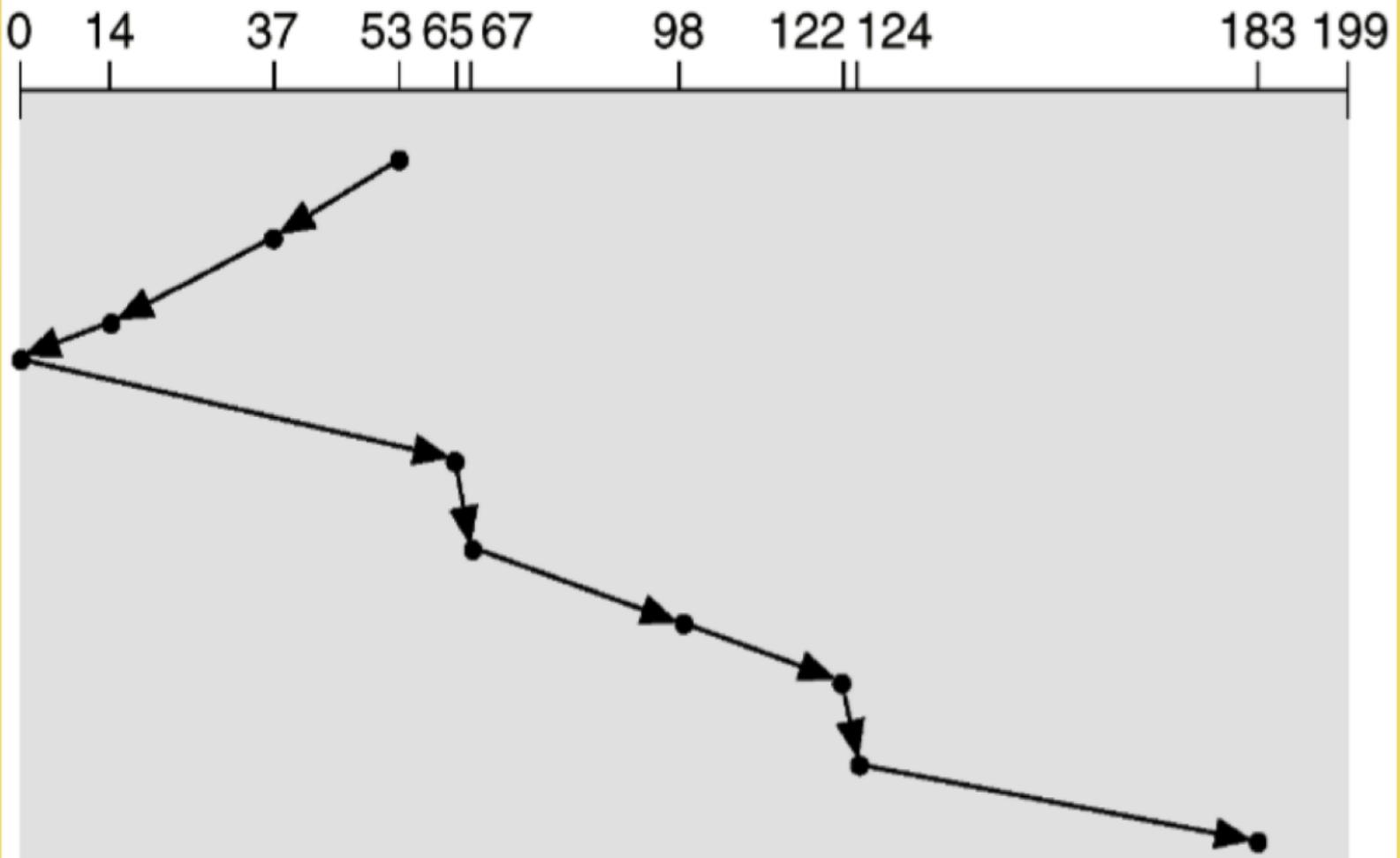


Shortest Time to Seek First



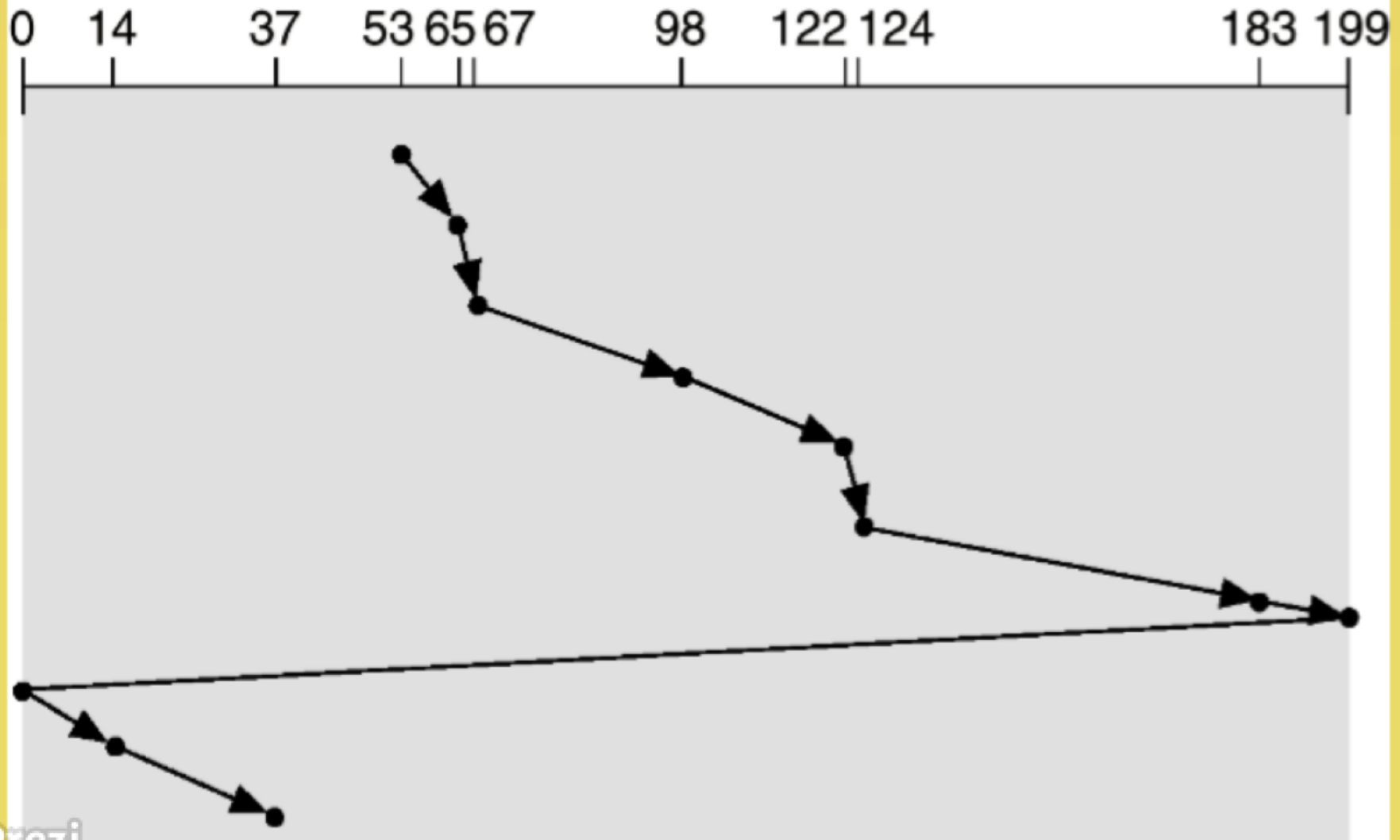
SCAN

queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53



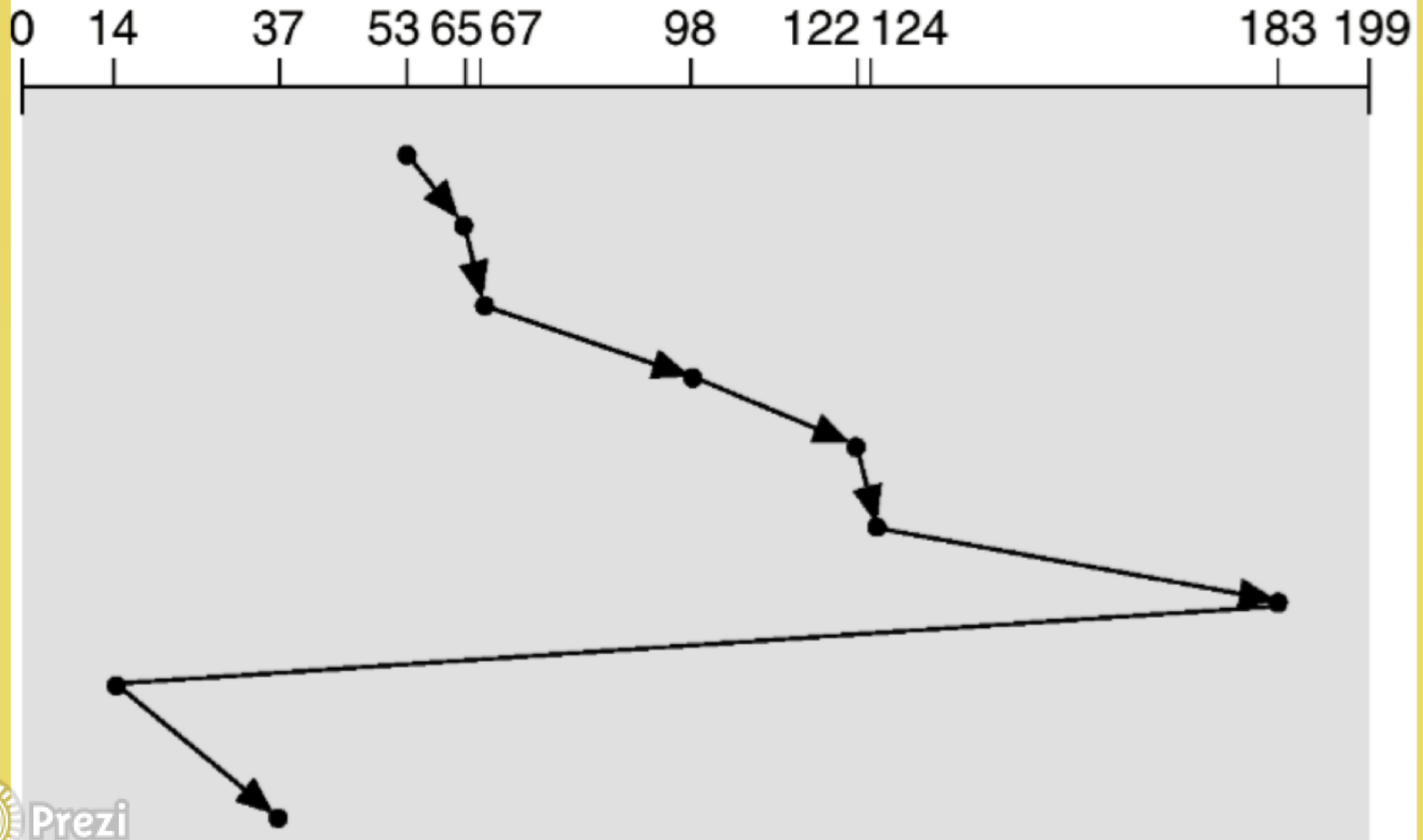
C-SCAN

queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53



C-LOOK

queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53



Sisteme de Fisiere Log-Structured

Scrie totul secvential

Tine minte unde este cea mai recenta copie pentru fiecare bloc

In paralel, sterge informatiile inutile

Consistenta unui sistem de fisiere

Pentru a scrie un singur fisier:

- scrie datele intr-un bloc sau mai multe
- scrie i-node-ul
- modifica lista de bloc-uri libere

Ce se intampla daca exista un defect?

Solutii posibile

FSCK: Exploateaza redundanta din FS

Teste de consistenta pentru blocuri:

- se construiesc table de blocuri libere si ocupate
- un bloc trebuie sa fie intr-un singur tabel

Teste de consistenta pentru fisiere:

- tabela cu numarul de referiri construita prin parcurgerea directoarelor
- se compara cu nr de referiri din i-node

Sisteme de fisiere jurnalizate

Idee:

- scrie modificarile pe disc inainte de a le aplica
- Aplica modificarile
- Sterge log-ul

La boot se verifica jurnalul si se efectueaza operatiile din tranzactii (trebuie sa fie idempotente)



FCK: Exploateaza redundanta din FS

Teste de consistenta pentru blocuri:

- se construiesc table de blocuri libere si ocupate
- un bloc trebuie sa fie intr-un singur tabel

Teste de consistenta pentru fisiere:

- tabela cu numarul de referiri construita prin parcurgerea directoarelor
- se compara cu nr de referiri din i-node

Sisteme de fisiere jurnalizate

Idee:

- scrie modificarile pe disc inainte de a le aplica
- Aplica modificarile
- Sterge log-ul

La boot se verifica jurnalul si se efectueaza operatiile din tranzactii (trebuie sa fie idempotente)

Avantaje? Dezavantaje?

timp scurt de verificare a consistentei
robustete

incetineste sistemul de fisiere

tica jurnalul si se efectuea
(trebuie sa fie idempotent

Avantaje? Dezavantaje?

timp scurt de verificare a consistentei
robustete

incetineste sistemul de fisiere

Concluzii

Fisierele reprezinta unitatea de stocare a informatiei

Sistemele de fisiere ofera o interfata comuna utilizatorilor pentru a lucra cu fisiere si directoare

Operatii principale:

- gestiunea datelor ce blocuri sunt intr-un fisier?
- organizare ierarhica: directoarele sunt fisiere speciale
- Gestiunea spatiului liber