Università di Roma Tor Vergata
Corso di Laurea triennale in Informatica

Sistemi operativi e reti

A.A. 2017-18

Pietro Frasca

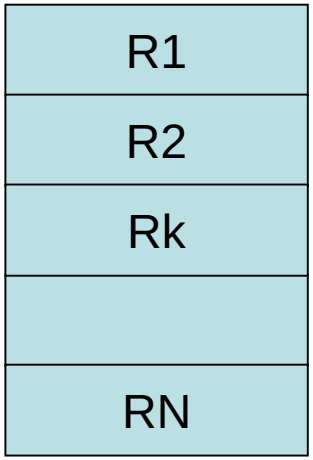
Lezione 22

Martedì 9-01-2018

Metodi di accesso

- I metodi di accesso determinano le operazioni che i processi possono eseguire sui file. I metodi di accesso più diffusi sono:
  - Accesso sequenziale
  - Accesso diretto (random)
  - Accesso indicizzato
- Ogni metodo di accesso dipende dalla particolare struttura del file, che a questo livello è visto come un insieme di record logici.

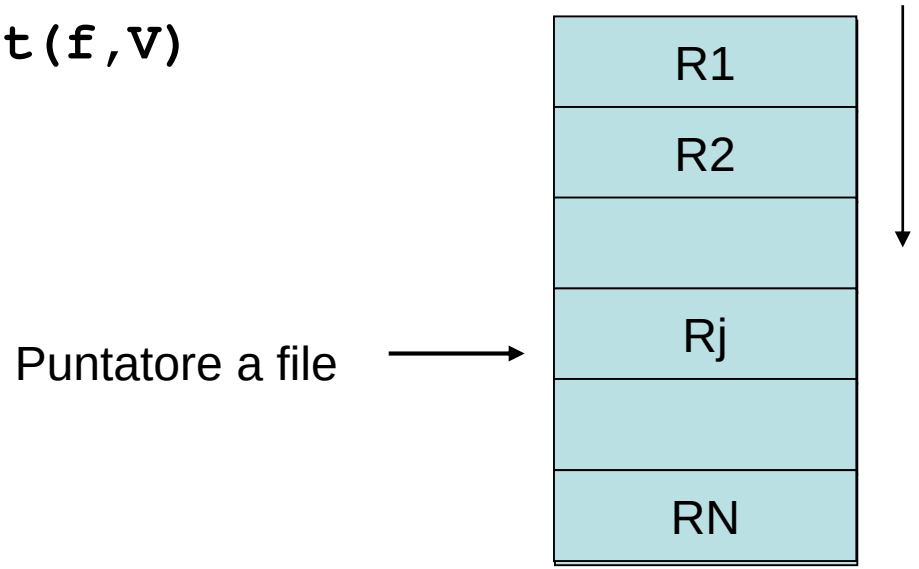
F: {R1, R2,…Rn}



- Il record logico è l'unità di lettura/scrittura sul file.

Accesso sequenziale

- Ogni file è formato da una sequenza di record logici.
- I record possono essere scritti e letti solo in sequenza, quindi, ad esempio, per leggere un record che si trova nella posizione j è necessario leggere tutti i j-1 precedenti.
- Il sistema utilizza un puntatore al file che indica il prossimo byte su cui leggere o scrivere.
- Le chiamate di sistema per l'accesso sequenziale sono del tipo:
  - **`readnext (f,V)`**
  - **`writenext(f,V)`**



- Per accedere al record j è necessario eseguire nel modo seguente:

```
for (i=1; i<j;i++)
  readnext(f, V); /* lettura dei record precedenti */
readnext(f ,V); /* lettura del record j */
```

Accesso diretto

- L’accesso diretto permette di leggere/scrivere un qualunque record del file con operazioni del tipo:

  - **readd(f, j, V)**
  - **writed(f, j, V)**

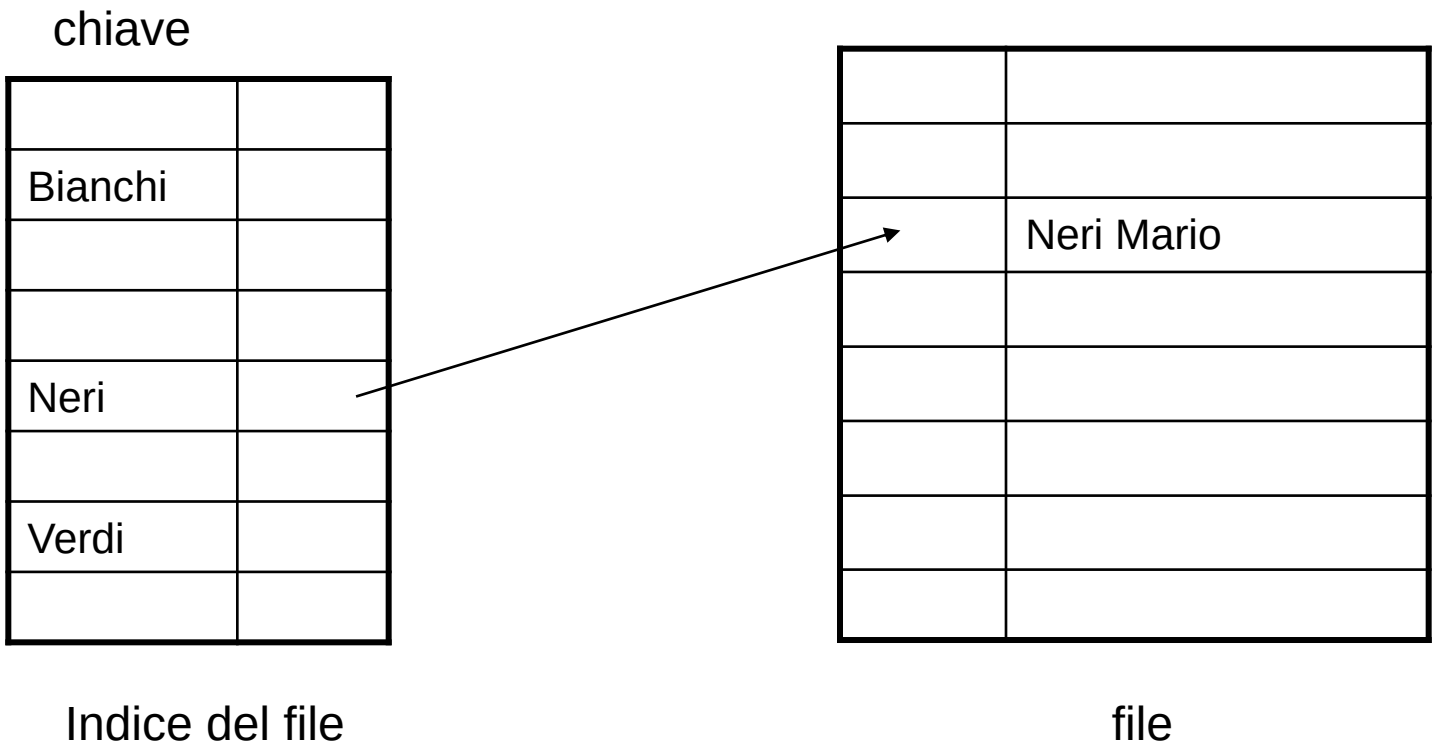
- Si può accedere ad un determinato record specificandone l’indice **j**. Non è necessario quindi dover scorrere i record precedenti.

Accesso a indice

Con l'accesso a indice la struttura del record logico è composta di almeno due campi, uno dei quali, la **chiave**, identifica univocamente un determinato record. Inoltre ad ogni file viene associata una tabella, detta **indice,** nella quale è presente una riga per ogni chiave che contiene un campo di riferimento al record corrispondente nel file. In questo modo è possibile accedere a un particolare record del file specificandone la chiave, mediante le operazioni:

- **readk(f, key, V)**
- **writek(f, key, V)**

La chiave consente di accedere direttamente al record cercato, previa ricerca associativa nell'indice del file.


chiave
Bianchi
Neri
Verdi
Neri Mario
Indice del file
file

Il livello organizzazione fisica

- In questo livello si implementano le tecniche per allocare i record logici nel dispositivo di memoria secondaria il quale è visto come insieme di **blocchi fisici**.
- Il blocco fisico (blocco) è l'unità minima di allocazione e di trasferimento dei dati. Nel caso dei dischi ogni blocco ha un suo indirizzo fisico.
- Non tutto lo spazio di un disco è usato per memorizzare i file. Alcune parti sono usate per mantenere la struttura logica del file system e per salvare le informazioni relative ai diritti di accesso dei file.

Tecniche di allocazione dei file

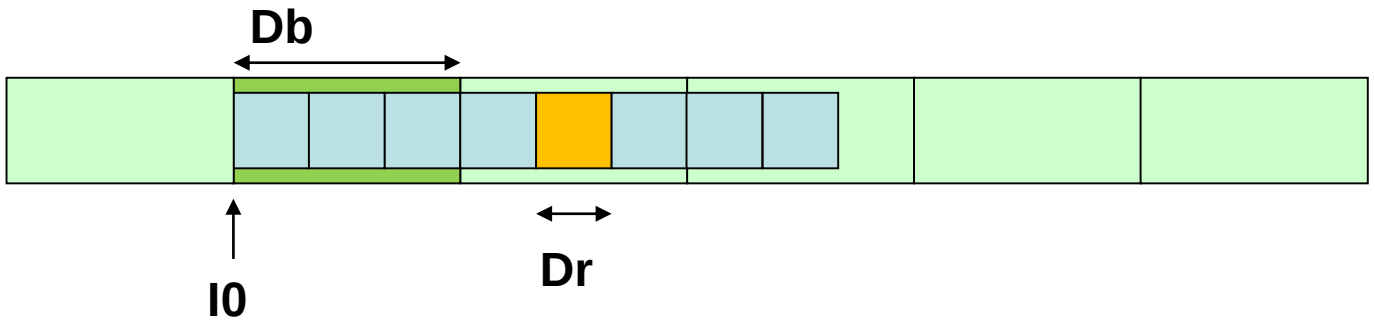
- Queste tecniche creano la corrispondenza tra i record logici contenuti in un file e l'insieme dei blocchi fisici in cui sono effettivamente memorizzati.
- I processi accedono ai file considerando il record logico come unità di accesso al file mentre l'unità di allocazione dei dati sul disco è il blocco (record fisico), la cui dimensione è fissa e tipicamente compresa tra 512 e 4096 byte.
- Ci sono varie tecniche con cui allocare i file, le più diffuse sono:
  - Allocazione contigua
  - Allocazione a lista
  - Allocazione a indice

Allocazione contigua

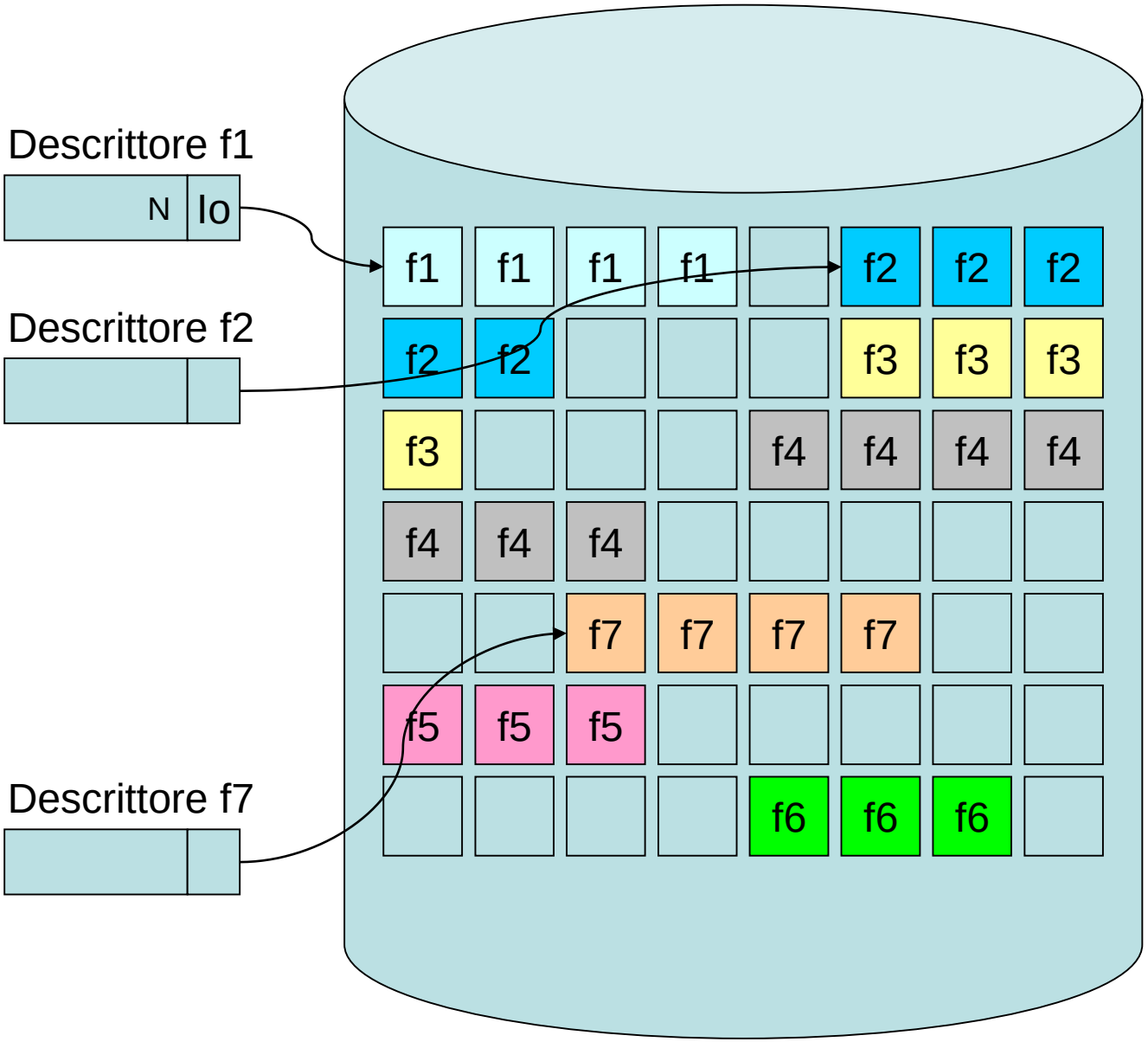
- Ogni file occupa un insieme di blocchi contigui. Con questa tecnica è semplice ed efficiente sia il metodo ad accesso sequenziale che il metodo ad accesso diretto.
- Il descrittore del file, contiene l'indirizzo **I0** del primo blocco in cui è allocato il file e il numero **N** di blocchi occupati. L'indirizzo del **blocco** in cui è allocato il record **Ri** è dato da:

  Ii=I0+i/NB

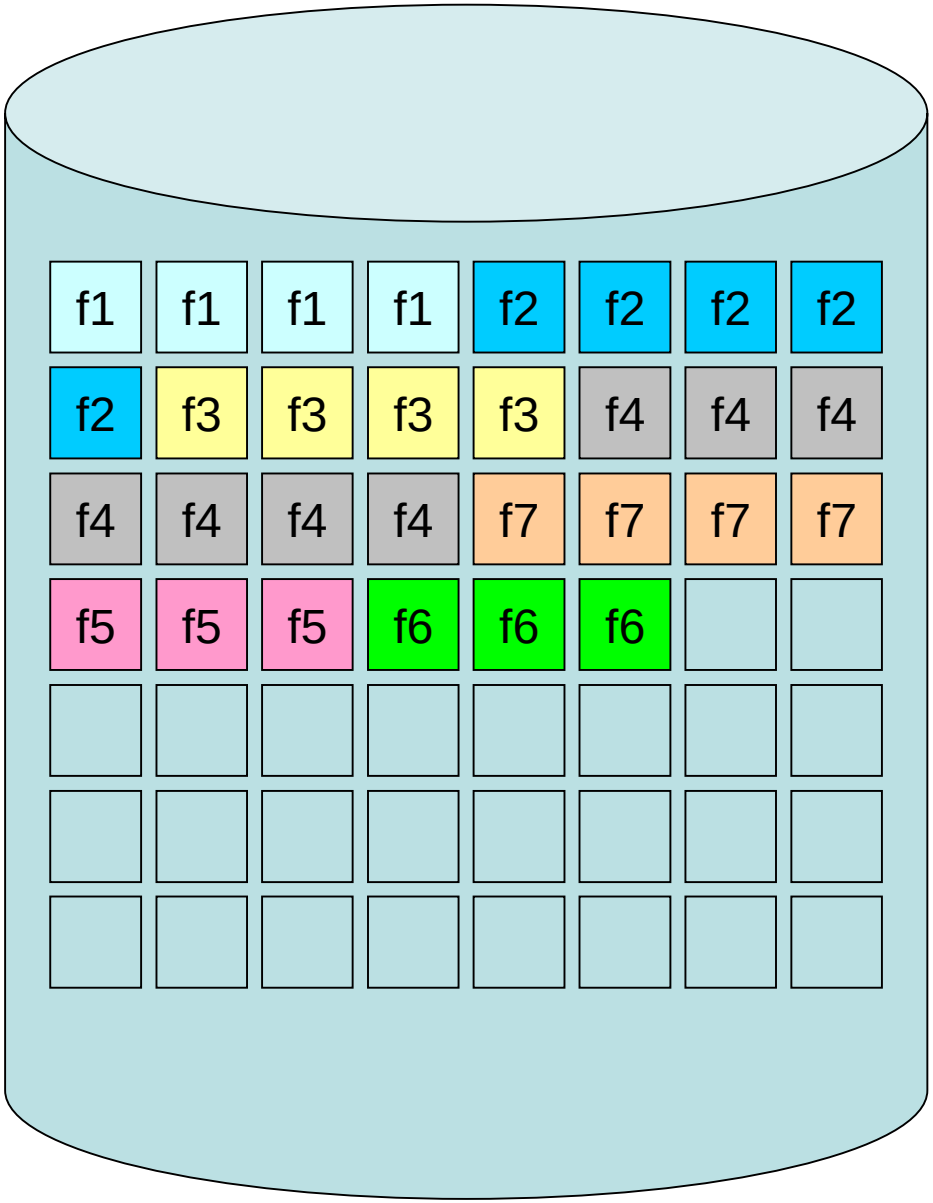
- Dove **NB = Db/Dr** è il numero di record logici contenuti in ogni blocco, avendo indicato con **Db** la dimensione del blocco e con **Dr** la dimensione del record logico.



- Il metodo di allocazione contigua presenta i seguenti **svantaggi**:
  - Ricerca difficile per trovare un insieme di blocchi adiacenti la cui dimensione sia sufficiente per contenere il file da allocare.
    Per la scelta dell'area contigua si ricorre a vari criteri, tra i quali:
    - **Best fit** – tende a minimizzare il numero di blocchi non utilizzati nell'area prescelta per allocare il file, scegliendo quindi l'area di dimensione più vicina (in eccesso) a quella del file.
    - **First fit –** viene scelta l'area di indirizzo (su disco) inferiore.
    - **Worst fit** – viene scelta la zona di estensione massima, favorendo in questo modo la possibilità di ulteriori allocazioni nella stessa zona.
  - Un altro svantaggio dell'allocazione contigua è che porta alla frammentazione del disco. Per risolvere questo problema si ricorre all'operazione di deframmentazione del disco che consiste nello spostare i file riallocandoli in modo contiguo eliminando così i numerosi buchi e ottenendo una zona libera contigua.



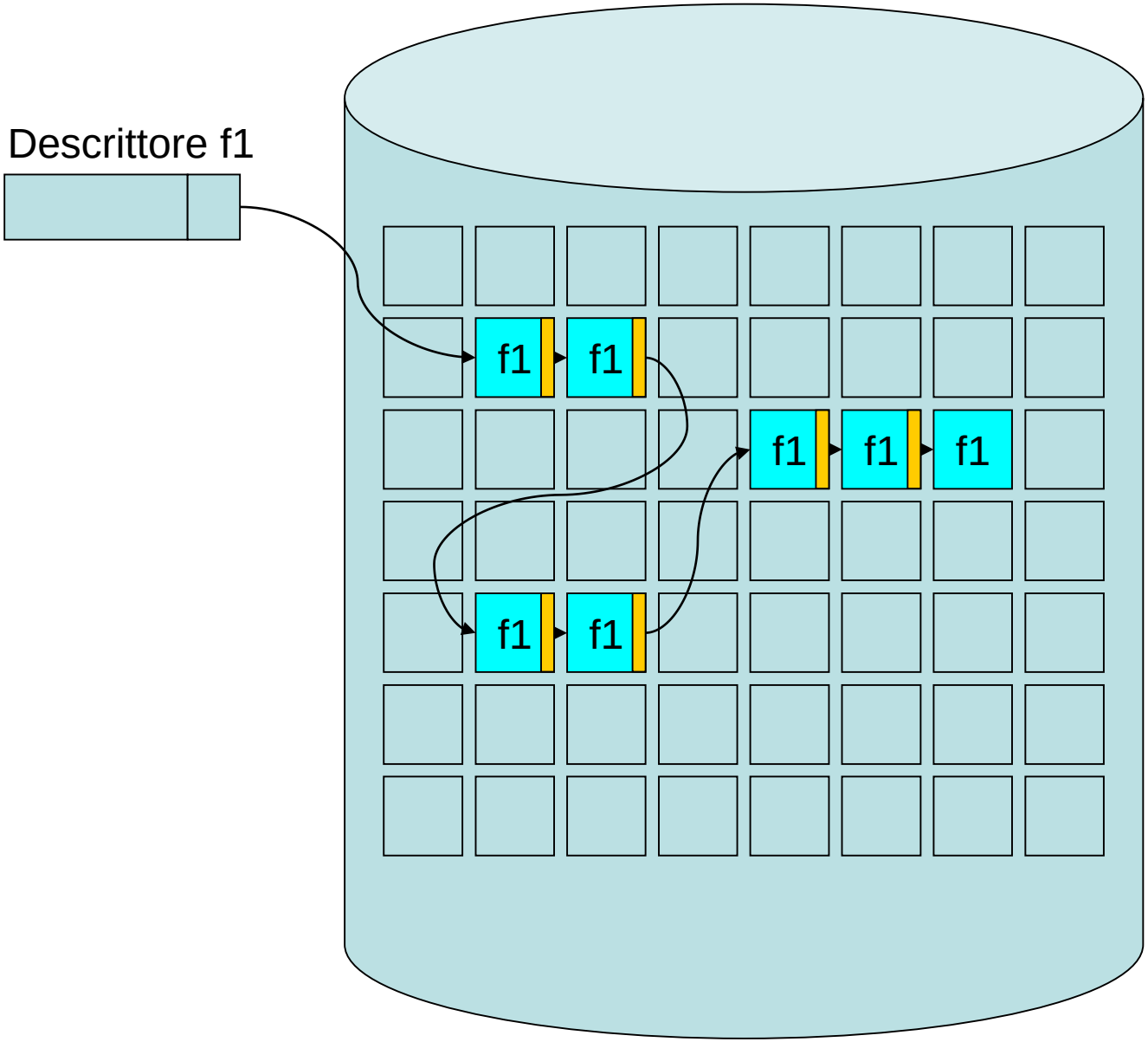
Allocazione contigua



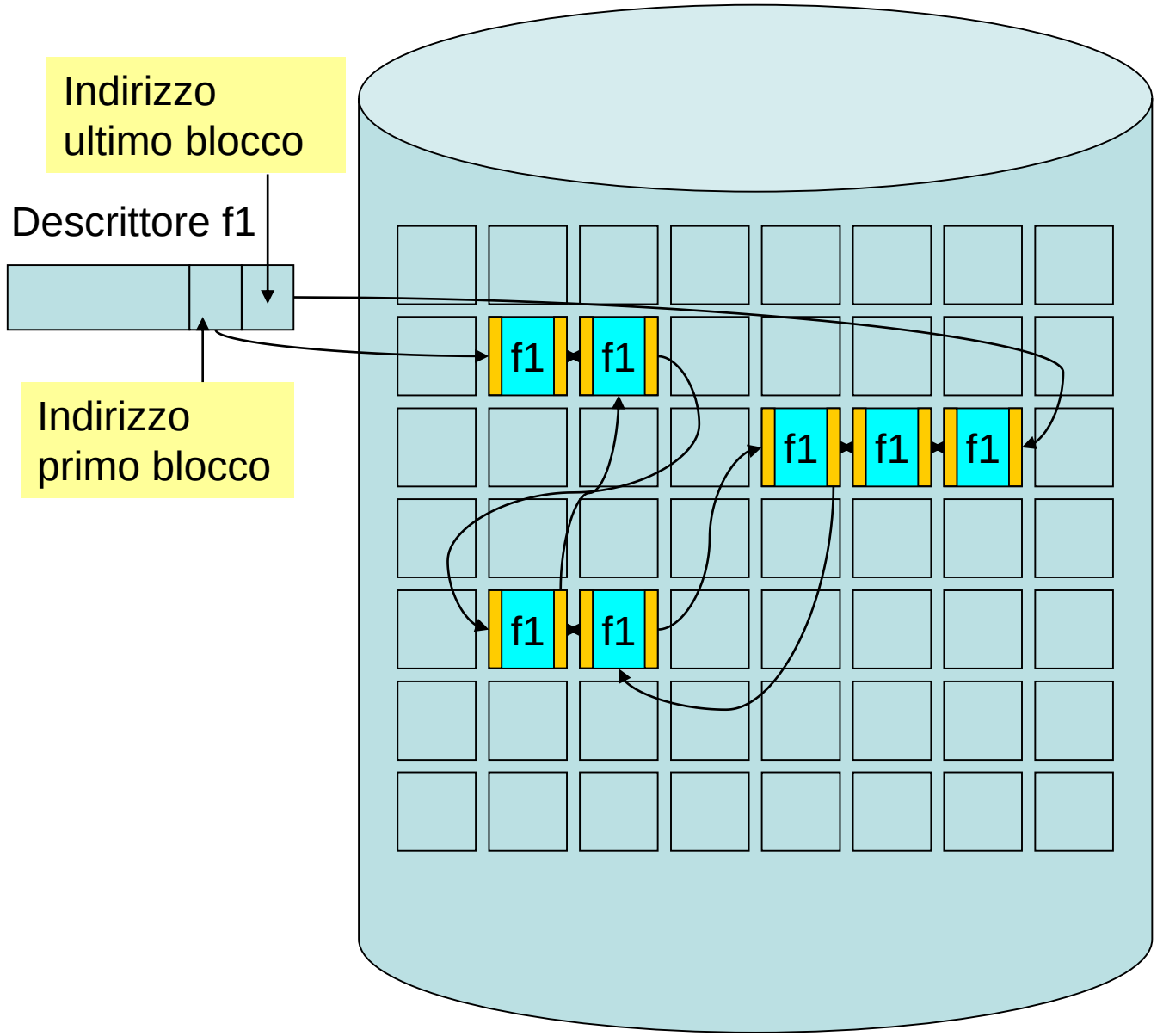
Deframmentazione del disco

Allocazione a lista concatenata

- Con questa tecnica il file è memorizzato in un insieme di blocchi non necessariamente adiacenti, organizzati in una lista concatenata.
- Il descrittore del file contiene il puntatore al primo blocco utilizzato. In ogni blocco è memorizzato, oltre ai dati, il puntatore al blocco successivo.
- Il metodo di accesso sequenziale è efficiente, mentre l'accesso diretto è invece lento, in quanto per accedere al blocco nel quale è allocato il record **Ri** sono necessari **i/Nb** (Nb = Db/Dr) accessi prima di arrivare al blocco desiderato.
- Questa tecnica ha il vantaggio di eliminare la frammentazione del disco eliminando il problema della ricerca dei blocchi liberi come nel caso dell'allocazione contigua. D'altra parte, le operazioni di accesso sono più lente, perché i blocchi sono sparsi sul disco.
- Il file system **è poco affidabile** poiché il guasto di un blocco implica la perdita del puntatore al blocco successivo, rendendo impossibile accedere all'intero file.

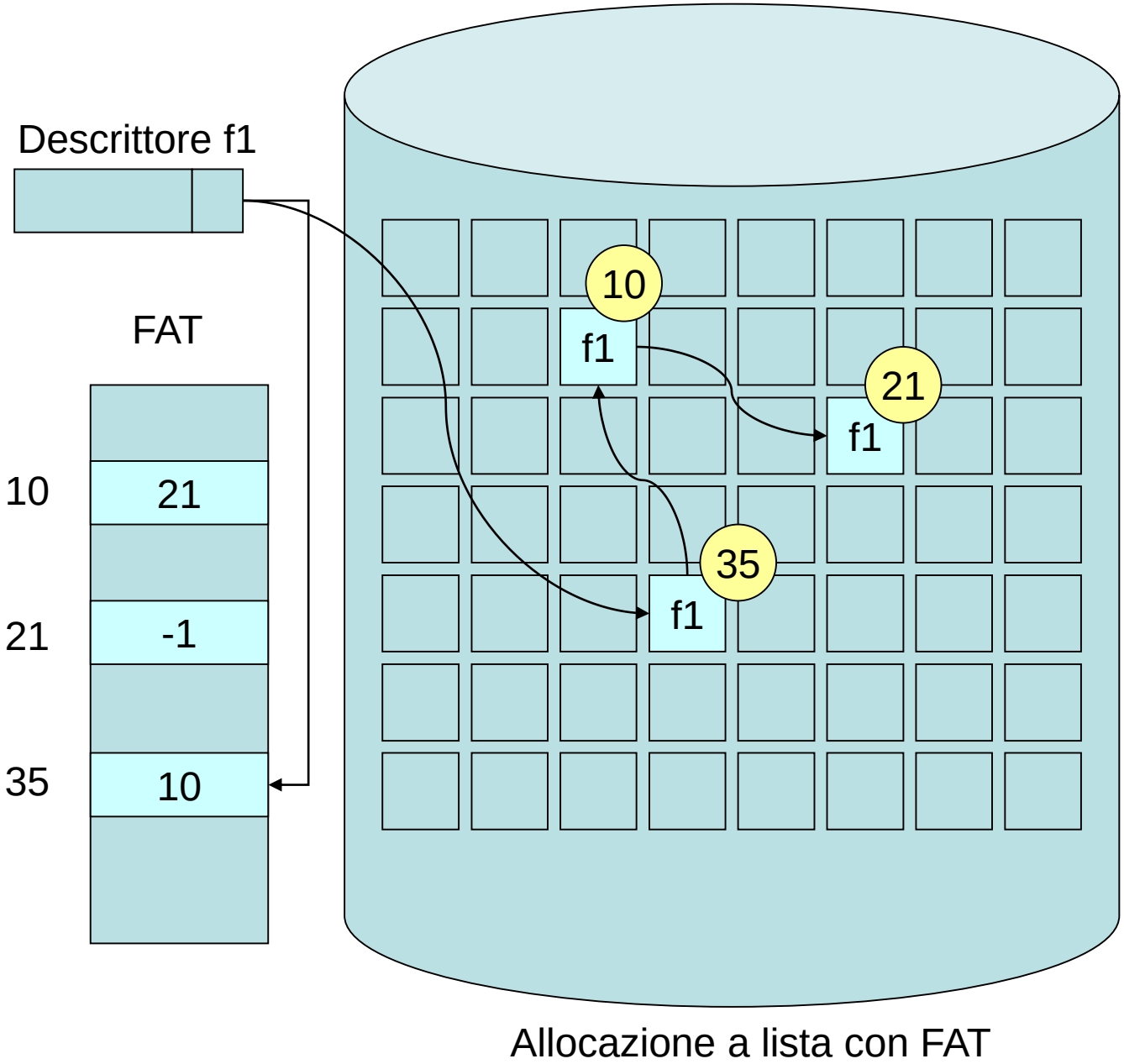


Allocazione a lista concatenata



Allocazione a lista con doppio collegamento

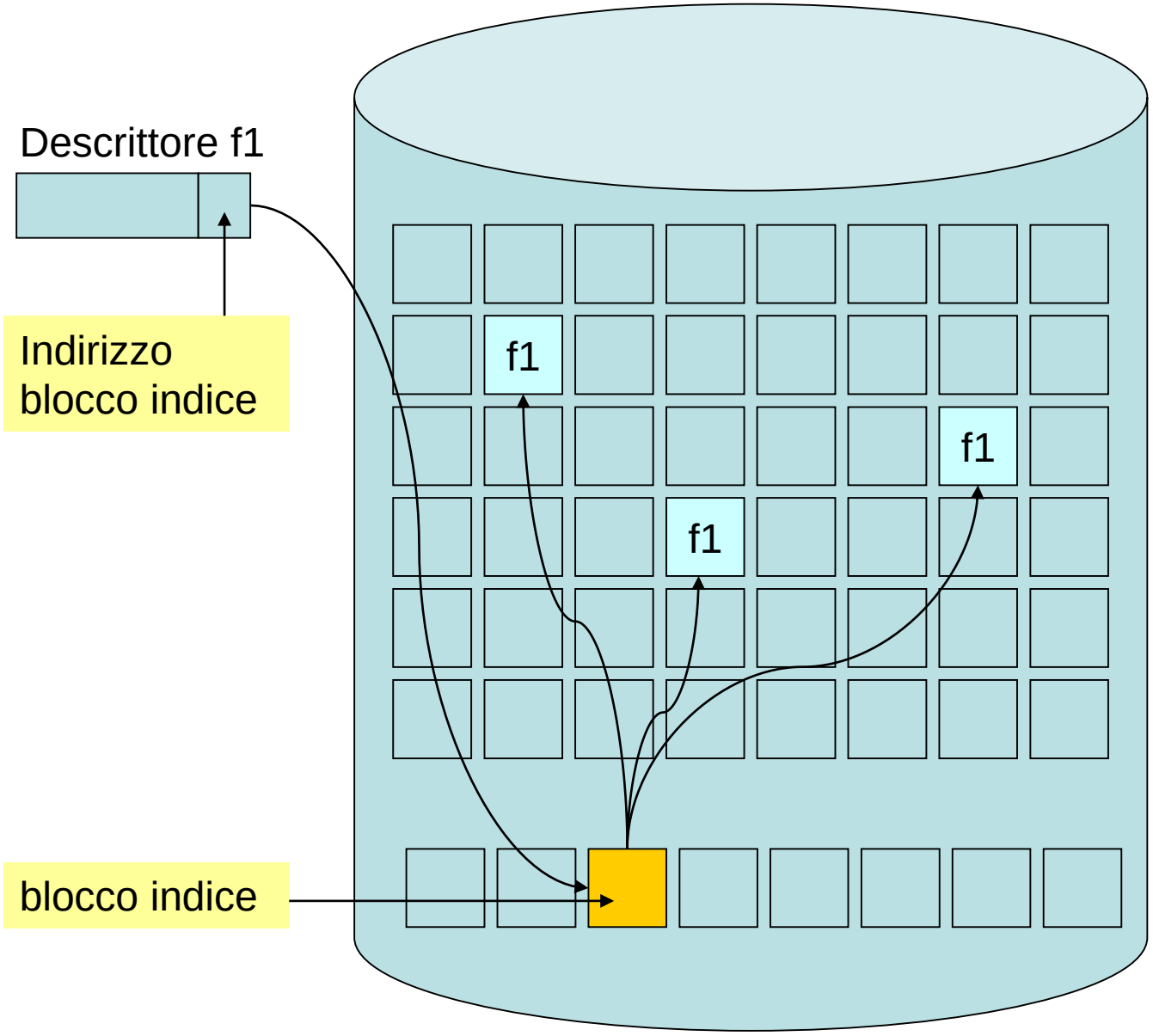
- Un miglioramento della tecnica a lista concatenata si ha utilizzando una struttura dati nella quale è descritta la mappa di allocazione di tutti i blocchi (**Tabella di allocazione dei file – File Allocation Table – FAT**).
  La FAT è memorizzata in una posizione fissa su disco (spesso è replicata su due zone del disco).
- La FAT contiene un elemento per ogni blocco del disco, il cui valore indica se il blocco è libero oppure contiene l'indice dell'elemento della tabella che rappresenta il blocco successivo nella lista.
- Se un puntatore si perde, si può ripristinare la concatenazione mediante la FAT. Copiando la FAT in memoria o in una cache si **velocizza notevolmente l'accesso diretto**, poiché l'indirizzo del blocco cui accedere può essere recuperato dalla RAM evitando continui accessi al disco.



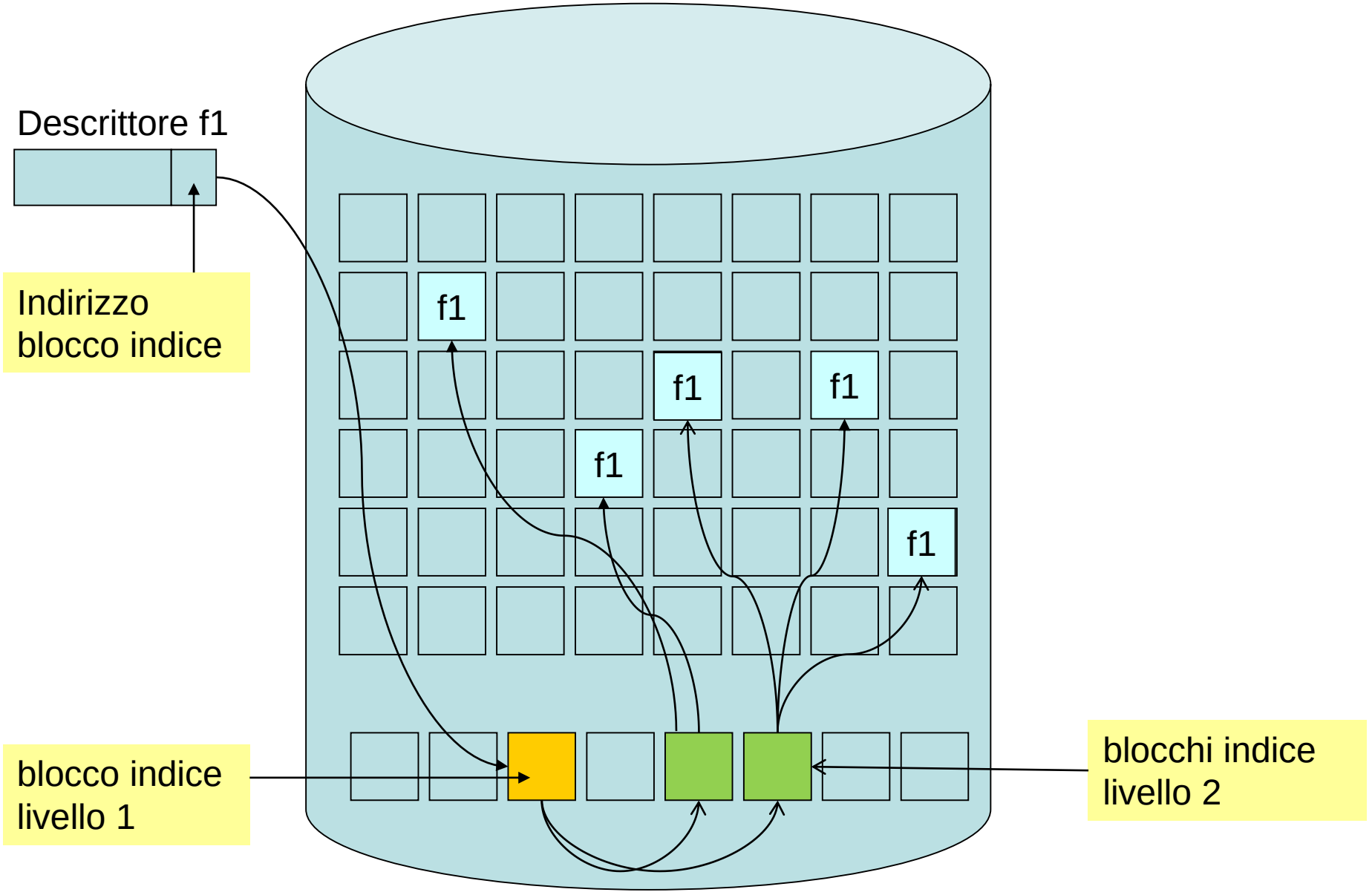
Allocazione a lista con FAT

Allocazione a indice

- Ad ogni file si associa un blocco indice che contiene gli indirizzi dei blocchi utilizzati per l'allocazione del contenuto del file.
- Il descrittore del file contiene l'indirizzo del blocco indice, accedendo al quale si ottengono gli indirizzi degli altri blocchi utilizzati per l'allocazione del file.
- Questo metodo è efficiente sia per l'accesso sequenziale che diretto.
- Uno svantaggio evidente è dato dalla limitata dimensione del blocco indice. Per eliminare questo limite si utilizzano vari livelli di indice. Ad esempio nel caso di indici a 2 livelli, ad ogni file è associato un indice di primo livello che contiene indirizzi di altri blocchi indice (di secondo livello), i quali contengono gli indirizzi dei blocchi utilizzati per l'allocazione dei file.
- Unix, ad esempio, utilizza un metodo a tre livelli di indice.



Allocazione a indice



Allocazione a 2 livelli di indice

Il livello dispositivo virtuale

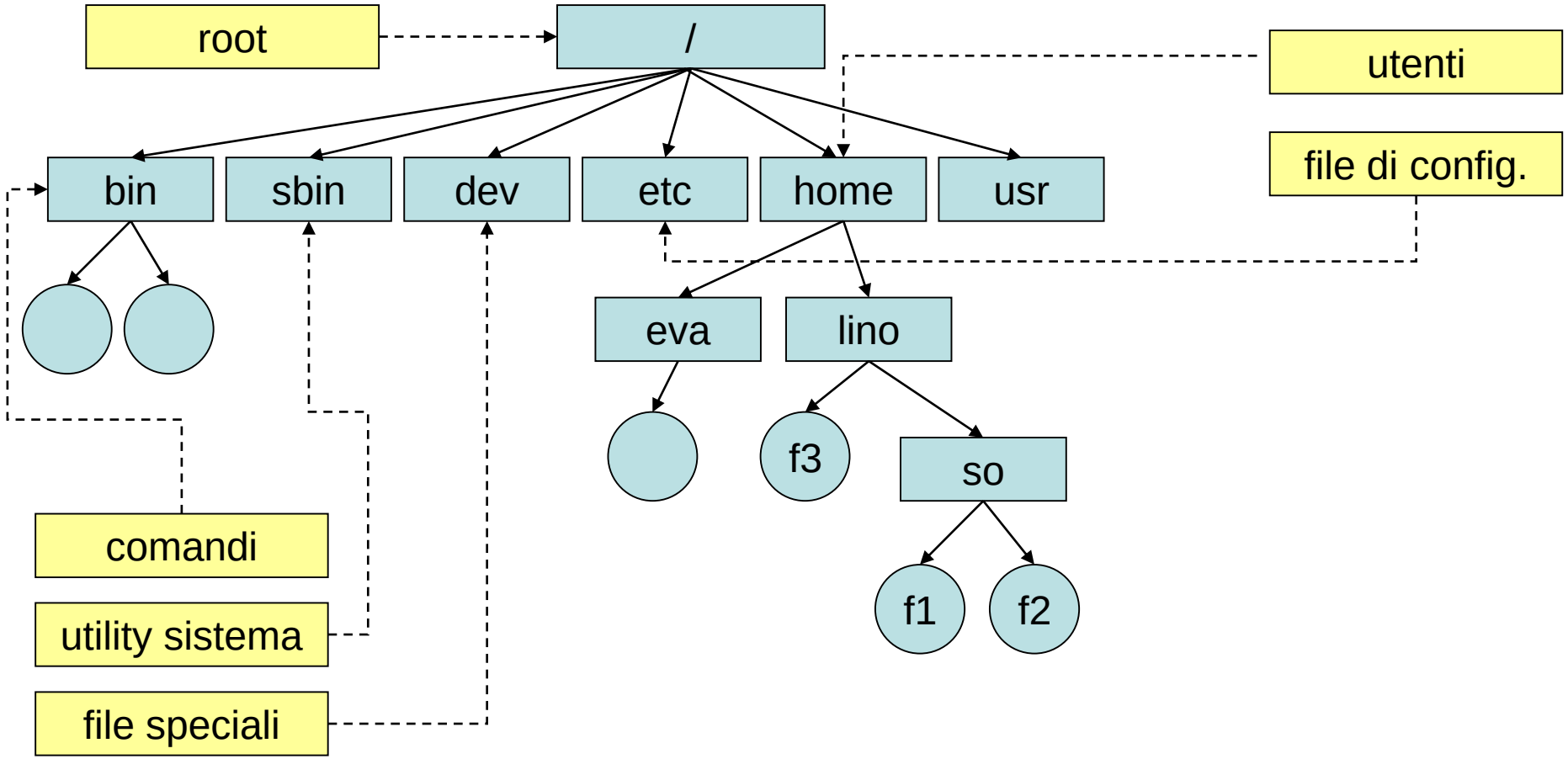
- Interagisce direttamente con l’hardware e ha il compito di **partizionare il disco** in un insieme di blocchi fisici di dimensione fissa, facendo vedere al livello soprastante il dispositivo come fosse costituito da un vettore lineare di blocchi fisici.

Il file system nei sistemi Unix

- Il file system si basa sulle astrazioni di file e directory. Il file è l'unità logica di memorizzazione dei dati, mentre la directory è la struttura che consente di raggruppare file e anche altre directory.
- Il file system di unix considera nello stesso modo sia risorse hardware che software. In particolare esistono tre tipi di file:
  - **Ordinario**
  - **Directory**
  - **Speciale**
- Il file speciale rappresenta un dispositivo fisico, come ad esempio un disco, una stampante, una porta seriale, etc.
- In unix tutti i file speciali sono memorizzati nella directory ***/*dev**

Struttura logica del file system

- una tipica organizzazione logica del file system di unix è mostrata nella figura seguente.
- La directory radice è indicata con il carattere ***/*** (barretta o slash).
- La navigazione nel file system, cioè l'operazione per passare da una directory corrente ad un'altra si ottiene mediante il comando **cd (change directory)**.
- A ogni shell in uso è associata una directory corrente che specifica la locazione corrente nel file system.
- I nomi dei file possono essere espressi in **formato assoluto** e in **formato relativo**. Il nome assoluto del file individua il percorso che è necessario compiere per giungere ad esso a partire dalla root. Il nome relativo indica il percorso che è necessario compiere a partire dalla directory corrente per arrivare al file.



Tipica organizzazione del file system di unix

- Ad esempio, in riferimento alla figura il nome del file assoluto di **f1** è **/home/lino/so/f1** mentre quello relativo, se la directory corrente è **lino**, è **so/f1**
- La directory corrente è indicata con il carattere **.** (punto), mentre la directory padre (parent) è indicata con **..** (punto punto). Quindi il nome relativo del file **f3**, supponendo che la directory corrente sia **so**, sarà **../f3**.
- La struttura del file system di unix è a **grafo aciclico**, dato che ad un file possono essere assegnati più nomi logici (file linkati). Il comando che consente di realizzare un link è **ln.**
- Ad esempio i seguenti comandi creano rispettivamente un link software e un link hardware al file **/usr/local/bin/pro1** assegnandogli i nomi **pro2** e **pro3** nella directory corrente.

ln –s /usr/local/bin/pro1 pro2

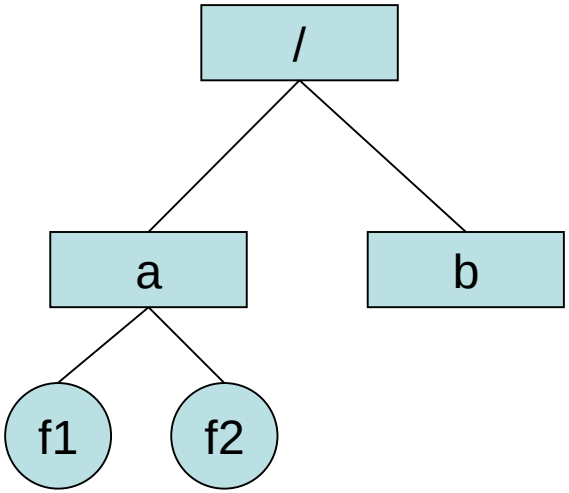
ln /usr/local/bin/pro1 pro3

- il nome di un file può quindi non essere unico, ma ad ogni file è associato un solo descrittore (chiamato **i-node**) che è univocamente identificato da un numero intero (detto **i-number**).
- La variabile di ambiente **PATH** indica la lista di directory nella quale deve essere ricercato il programma che si vuole eseguire. Essa è una stringa formata da un sequenza di directory, ciascuna delle quali è separata dalla successiva da un carattere separatore (carattere **:**). Ad esempio

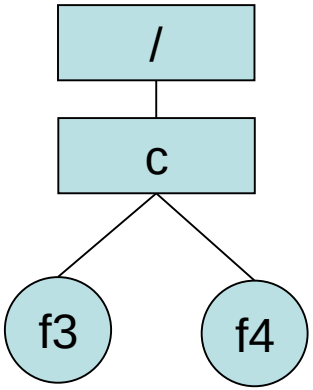
  PATH="/bin:/usr/local/bin:."

- Unix permette di montare un disco nel file system di un altro disco. Nell'esempio in figura il file system del disco B è montato sulla directory b del disco A. Il comando è **mount**.

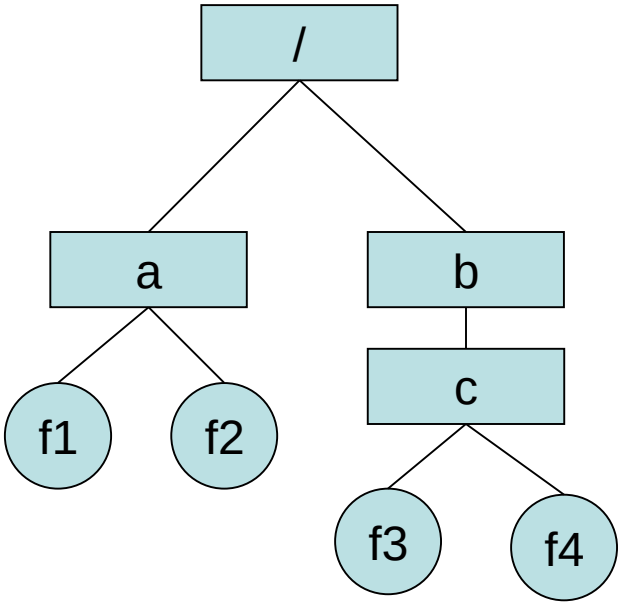
Disco rigido



CD B



Disco rigido + CD



Quindi per copiare il file f3 contenuto nel CD, dopo il mount si può fare:

cp /b/c/f3 .