

D B Levyn rakenne

- Levykössä (disk drive) on useita samankeskisiä levyjä (disk)
- Levyissä on magneettinen pinta (disk surface) kummallakin puolella levyä
- Levyllä on osoitettavissa olevia uria (track), muutamasta sadasta muutamaan tuhanteen, päällekkäiset urat muodostavat sylinterin (cylinder)
- Urat voivat jakautuvat sektoreihin (sector)

1

D B Levykön rakenne

2

D B Levyn rakenne

- Lohko (=levyjakso, block) on yhdestä tai useammasta peräkkäisestä sektorista muodostuva alue uralla.
- Lohkokoko on yleensä 512 tavun monikerta (kakkosen potenssi). Lohkot erotetaan toisistaan lohkovälillä, johon talletetaan lohkon liittyvää kontrollitietoa.
 - jos sektorikoko on 512B ja lohkokoko 8KB muodostuu yksi lohko 16 peräkkäisestä sektorista
- Tietoa siirretään levyltä ja levyille aina kokonaisina lohkoina.
 - lohkon fyysinen osoite muodostuu **levypinnan numerosta, uran numerosta** ja uran sisäisestä **lohkon numerosta**

3

D B Levyn rakenne

- Lohkon sisällön hakemiseksi
 - Siirretään luku/kirjoituspäät halutulle sylinterille (pääät siirtyvät kullakin pinnalla samanaikaisesti yhdessä)
 - Odotetaan, että haluttu lohko pyörähtää luku/kirjoituspään kohdalle (levyn pyörimisnopeus luokkaa 100 kierrosta sekunnissa)
 - Aktivoidaan halutulla pinnalla oleva luku/kirjoituspää lukemaan dataa – yleensä vain yksi pää voi olla samanaikaisesti aktiivinen
 - I/O prosessori siirtää luetun datan hakupyynnön yhteydessä annettuun puskuriin

4

D B Hakuaika

- Levyiltä hakemiseen kuluva aika (hakuaika, access time) muodostuu:
 - hakuvarren siirtoajasta eli kohdistusajasta (seek time) (sylinterille siirtyminen) (*st*)
 - aika riippuu siitä millä sylinterillä päät ovat alunperin, lyhyt siirtymä vie vähemmän aikaa, 0 – levykohtainen maksimi. Usein ilmoitetaan keskimääräinen kohdistusaika esim. 6 ms
 - pyörähdysviiveestä (rotational delay) (*rd*)
 - riippuu alkuperäisestä kohdasta ja pyörimisnopeudesta.
 - lohkon siirtoajasta (block transfer time) (*btf*)
 - riippuu uran kapasiteetista, pyörimisnopeudesta ja lohkon koosta


5

D B Pyörähdysviive

- Jos levyn pyörimisnopeus on p kierrosta minuutissa (rpm), niin keskimääräinen pyörähdysviive


$$rd = \frac{1}{2} * (1/p) \text{ min} = \frac{1}{2} * (60 * 1000 / p) \text{ ms}$$
- Jos $p=10000$, niin $rd=3 \text{ ms}$ (tyypillinen)
- Jos $p=6000$, niin $rd= 5 \text{ ms}$

6

 Lohkon siirtoaika


- Lohkon siirtoaika (= aika, joka kuluu siihen, että lukupää kulkee lohkon sektoreiden yli = täyden kierroksen pyörähdysaika kerrottuna lohkon koon suhteella uran kapasiteettiin
- Esimerkki:
 - Olkoon lohkokoko 4KB, uralle mahtukoon 50 lohkoa, joten uran kapasiteetti on 200KB. Olkoon kierrosnopeus 6000 rpm eli 100 kierrosta sekunnissa. Tällöin siirtonopeus (transfer rate, tr) = $100 \cdot 200 \text{ KB/s} = 20 \text{ KB/ms}$ ja lohkon siirtoaika = lohkokoko/siirtonopeus = 0.2 ms

7

 Haku aika


- Mitä lähempänä (oikeaan suuntaan) nykyistä kohtaa seuraava haettava jakso on sitä nopeampaa on haku, nopeasta hitaaseen
 - seuraavana samalla uralla
 - samalla sylinterillä 'seuraavana' – ei pyörähdysviivettä
 - jossain samalla uralla tai sylinterillä
 - läheisellä sylinterillä
 - kaukaisella sylinterillä

8

 Haku aika


- Peräkkäisten saman uran ja sylinterin jaksoiden siirtäminen on tehokkainta koska tällöin ei tule hakuvarren siirtoa eikä pyörähdysviivettä.
- Levyjen siirtonopeudet ilmoitetaan usein bitteinä sekunnissa. Voidaan antaa joko sisäinen siirtonopeus ja tehollinen siirtonopeus

9

 Haku aika


- Olkoon
 - st = kohdistusaika
 - rd = pyörähdysviive
 - btt = jakson siirtoaika = kierrosaika/uran lohkojen lukumäärä.
- Lohkon haku aika = $sd + rd + btt$
 - kohdistusaika ja pyörähdysviive vaihtelevat, arvioissa käytetään usein keskimääräistä haku aikaa
- Sivupyynnön toteutuksen palvelu aikaan on vielä lisättävä levyohjaimen jonotusaika sillä levyohjain voi suorittaa vain yhtä pyyntöä kerrallaan – jonotusaikaan vaikuttaa kaikki levyille kohdistuva kuorma

10

 Usean lohkon haku aika


- Haettaessa k lohkoa, lohkojen yhteenlaskettu saantiaika on
 - enintään $k \cdot (st + rd + btt)$, jos lohkot sijaitsevat satunnaisesti levyllä
 - $sd + k \cdot (rd + btt)$, jos lohkot sijaitsevat satunnaisesti samalla sylinterillä ja
 - minimissään $sd + rd + k \cdot btt$, jos lohkot ovat peräkkäin samalla sylinterillä
- Kaavojen tekijöistä sd ja rd ovat samaa millisekunneissa mitattavaa kokoluokkaa ja btt T noin kymmes- - tuhannesosa niistä (millisekunnin osia)
- Jos $S=5\text{ms}$, $R=5\text{ms}$ ja $T=0.2 \text{ ms}$ ja $k=1000$, niin kokonaissaantiajan vaihteluväli olisi 0.22s - 10.2s

11

 Levytiedoston käsittely


- Tiedostot sijoitetaan levyille
 - yhdelle tai useammalle peräkkäisistä lohkoista koostuvalle alueelle
 - voidaan hyödyntää peräkkäiskäsittelyn nopeutta koko tiedoston lukemisessa
 - puskurien hallinnassa voidaan käyttää ns. ennakoivaa haku (pre-fetch) – haetaan valmiiksi puskurisiin järjestyksessä seuraavia sivuja vaikka niitä ei ole vielä pyydetty

12

 **Levytiedoston käsittely**


- Vaikka tiedoston tietueet alunperin sijoitettaisiin tiiviisti lohkoihin voivat poistot ja muutokset huonontaa rakennetta
 - peräkkäisjärjestykseen jää väliin tyhjiä lohkoja poistojen takia
 - jatkotietueet ylivuotolohkoissa vaativat poikkeamista optimaalisesta lukemisjärjestyksestä – mahdollisesti jopa siirtymistä pois sylinteriltä

13

 **RAID-levyt ja rinnakkaisuus**


- RAID = redundant array of independent disks
- ryhmä toisistaan riippumattomia levyjä, jotka toimivat rinnakkain yhtenä loogisena levynä
- Kullakin levyllä oma levyohjain
- tavoitteena luotettavuuden ja tehokkuuden parantaminen
 - tehokkuutta parannetaan viipaloinnilla (striping)
 - luotettavuutta redundanssilla (ylimäärällä)
- Nykyaikainen levyratkaisu

14

 **RAID-levyt ja rinnakkaisuus**


- Viipaloinnin periaatteena on jakaa tiedosto usealle levyille. Tiedonsiirtoa levyjen ja puskureiden välillä voidaan suorittaa rinnakkain koska jokainen levyohjain toimii itsenäisesti. Näin tiedonsiirtonopeus kasvaa.
- Käytössä on eri viipalekokoihin (striping unit) perustuvia viipalointiperiaatteita. Jos käytettävissä on D levyä, sijoitetaan i:s viipale levyille $i \bmod D$.

15

 **RAID-levyt ja rinnakkaisuus**

- Viipalekoko voisi olla yksi bitti.
 - Tällöin D peräkkäistä bittiä ovat kukin eri levyillä ja kaikki D levyä osallistuvat jokaiseen hakuoperaatioon.
 - Koska pienin siirtoyksikkö on lohko joudutaan sivupyynnön yhteydessä siirtämään D lohkoa.
 - Jos käsittely on peräkkäiskäsittelyä, on siirron yhteydessä tehty ennakoiva haku D-kertaisella siirtonopeudella
 - Jos käsittely on hajakäsittelyä, on siirretty turhaa dataa, mutta siihen ei ole mennyt enempää aikaa kuin yhtä levyä käytettäessä


16

 **RAID-levyt ja rinnakkaisuus**

yhden bitin viipalekoko – 4 levyä



0110100100110010001111

17

 **RAID-levyt ja rinnakkaisuus**



- Lohkotason viipaloinnissa viipaleen koko on yksi lohko.
 - Jos sivupyynnöt kohdistuvat eri levyille voidaan pyynnöt suorittaa rinnakkain. Jonotusaika pienenee joten myös hajakäsittely nopeutuu.
 - Peräkkäiskäsittelyssä peräkkäiset lohkot sijaitsevat eri levyillä, joten tiedonsiirtonopeus kasvaa
 - Suurella levymäärällä saadaan suuremmat tehot

18

RAID-levyt ja rinnakkaisuus



- Tietojen jakaminen usealle levyille aiheuttaa luotettavuusongelman.
- N levyn järjestelmässä häiriön todennäköisyys on N-kertainen yhteen levyyn verrattuna
- Jos yhden levyn kohdalla häiriövälin odotusarvo on 50000 tuntia (5.7 vuotta), niin 100 levyn järjestelmässä se onkin enää $50000/100 = 500$ tuntia eli noin 21 päivää
- Luotettavuutta lisätään ylimäärällä, jolloin häiriöitä voi sietää.

19

RAID-levyt ja rinnakkaisuus

- Eräs ylimäärän muoto on peilaus (mirroring)
 - Tieto kirjoitetaan kahdelle identtiselle levyille.
 - Jos toinen levy vikaantuu, voidaan jatkaa toisella kunnes vika saadaan korjattua.
 - Kumpaakin levyä voi käyttää sivupyynnöiden toteutukseen, pyyntö ohjataan levyille, jolla on lyhyempi saantiaika.
 - Levyjen tehollinen kapasiteetti puolittuu
 - Kallis ratkaisu

20

RAID-levyt ja rinnakkaisuus

- Peilauksen vaihtoehtona on erilaisia virhekorjauksen mahdollistavia tekniikoita: Hamming koodi, bittitason tai lohkotason viipalointiin perustuvat pariteettibitit omalla levyllään tai hajautettuna yksikön levyille.

21