# Salvestussüsteemid



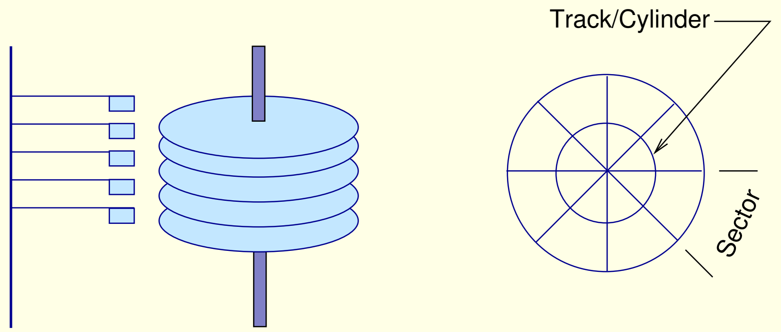
## Ülevaade

- Ketta struktuur
- Kettapöörduste planeerimine
- Ketaste ettevalmistamine
- Saalimisala haldus
- RAID struktuurid
- Stabiilne salvestusruum





## Ketta struktuur








## Ketta parameetrid

| Parameter | IBM 360-KB floppy disk | WD 18300 hard disk |
|---|---|---|
| Number of cylinders | 40 | 10601 |
| Tracks per cylinder | 2 | 12 |
| Sectors per track | 9 | 281 (avg) |
| Sectors per disk | 720 | 35742000 |
| Bytes per sector | 512 | 512 |
| Disk capacity | 360 KB | 18.3 GB |
| Seek time (adjacent cylinders) | 6 msec | 0.8 msec |
| Seek time (average case) | 77 msec | 6.9 msec |
| Rotation time | 200 msec | 8.33 msec |
| Motor stop/start time | 250 msec | 20 sec |
| Time to transfer 1 sector | 22 msec | 17 $\mu$sec |

# Kettapöörduste planeerimine

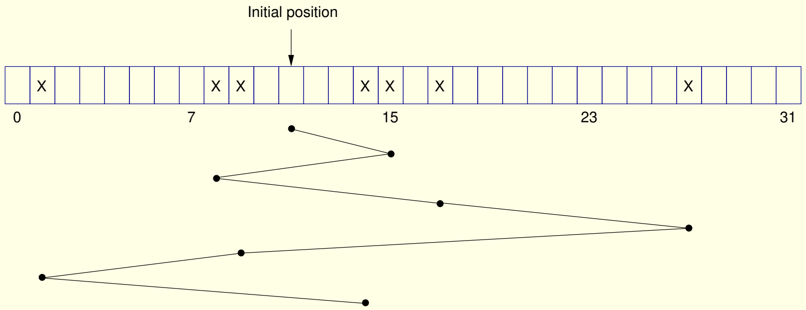
- Opsüsteem vastutab kettapöörduste efektiivsuse eest
  - Läbilaskekiirus suureks
  - Viivitus väikseks
- Pöördusajal kaks olulist tegurit:
  - Raja otsimine — aeg sõltub selest, kuipalju on pead vaja liigutada
  - Rajalt sektori otsimine (keskmiselt pool pööret)
- Läbilaske maksimaalne kiirus sõltub kettaseadmest
- Opsüsteem tahab positsioneerimise välja optimeerida, et läbilase oleks maksimumi lähedane





# Kettapöörduste planeermise algoritmid

- FCFS — *First Come First Served* (69)



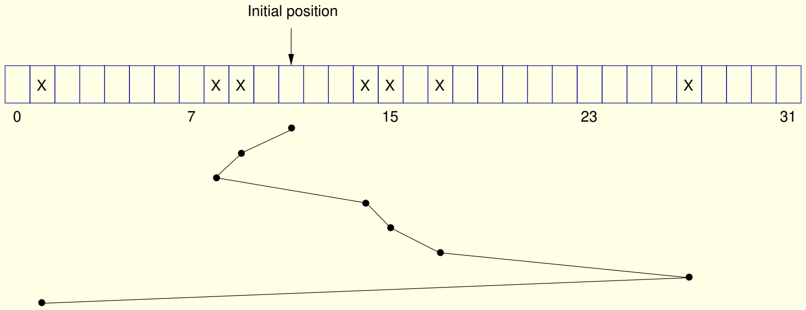

- Näide: päringute järjekord 15, 8, 17, 27, 9, 1, 14
- Kokku 69 raja vahetust





# Kettapöörduste planeermise algoritmid

- SSTF — *Shortest Seek Time First* (48)



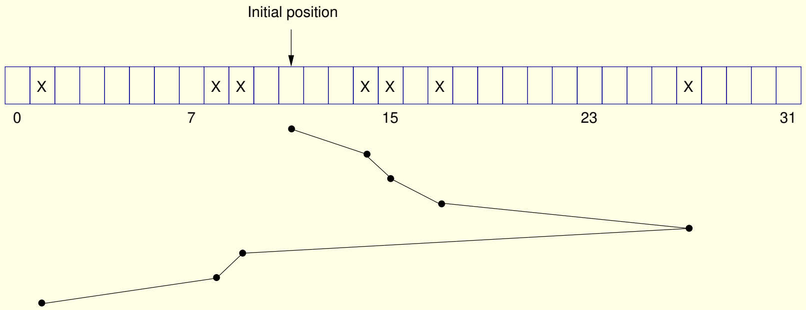

- Sarnane SJF protsesside planeerimisalgorimiga
- . . . kuid pole optimalne
- Võib põhjustada näljutamist





# Kettapöörduste planeermise algoritmid

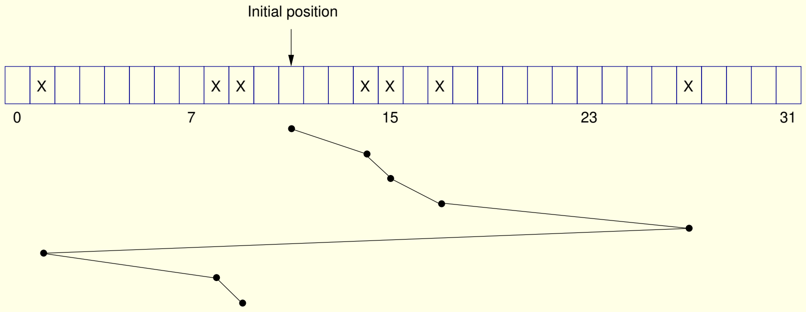
- SCAN (ka elevaatoralgoritm) — pea käib ketta ühest äärest teise ja teenindab teele jäävaid päringuid (50)
- LOOK — pea viiakse kummaski otsas ainult äärmise päringuni (42)




- Keskel asuvaid sektoreid teenindatakse kiiremini

# Kettapöörduste planeermise algoritmid

- C-SCAN — nagu SCAN, aga tagasiteel päringuid ei teenindata (61)
- C-LOOK — pea viiakse otstes ainult äärmise päringuni (51)




- Päringute teenendamise ooteajad on ühtlasemad





# Ketaste ettevalmistamine

- Madala taseme (füüsiline) formaatimine (*low level formatting*)
  - Sektorimärkide radadele kandmine
- Partitsioneerimine
- Loogiline formaatimine ehk failisüsteemi tekitamine
- Bootloaderi installimine
- Vigaste kettaplokkide leidmine ja meelde jätmine





# Madala taseme formaatimine

- Rada koosneb sektoritest, millede vahel on eraldajaks väiksed augud
- Sektori struktuur:

| | | |
|---|---|---|
| Preamble | Data | ECC |

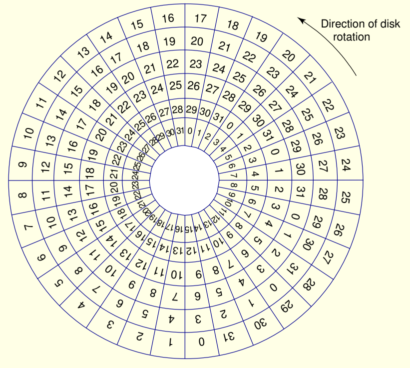
- Päis sisaldab:
  - Sektori algust näitavast bitimustrist
  - Silindri ja sektori numbrist
  - ...
- Andmepiirkonna suuruseks reeglina 512 baiti
- Veakontrollkood enamasti 16 baiti





# Madala taseme formaatimine

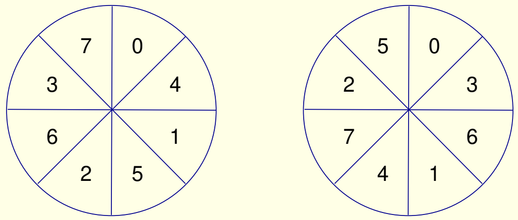
- *Cylinder skew* — erinevate radade sektorid on üksteise suhtes nihkes




- Aitab vähendada roteerumisviivitust järjestikuste sektorite lugemisel, mis paiknevad erinevatel radadel

# Madala taseme formaatimine

- *Interleaving* — sektorid paigutatakse üle ühe



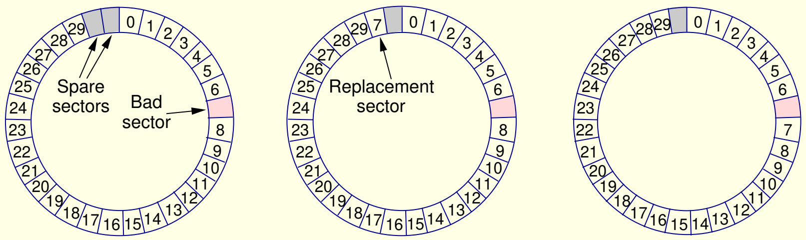
- Aitab vältida olukorda, et eelmise sektori andmete ülekandmise ajal liigub lugemispea üle järgmise sektori
- Aeglasema andme ülekandmise korral kasutusel ka üle kahe paigutus (*double interleaving*)
- Pole vaja kasutada, kui kontroller puhverdab kogu raja tervikuna





# Veatöötlus

- Tavaliselt jäetakse igale rajale mõned sektorid vabaks
- Kui mõni sektor muutub vigaseks, siis ta asendatakse vaba sektoriga
- Alternatiiv — sektorid nihutatakse vigasest sektorist ühe võrra edasi








# Saalimisala haldamine

- Saalimisala — protsesside või lehekülgede salvestusruum virtuaalmälule
- Võib paikneda kas eraldi kettal/partitsioonil või eelallokeeritud failis
- RAM-is peab saalimisala kohta olema kaart
- BSD UNIX — saalimisala reserveeritakse protsessi käivitamisel
  - Eraldi saalimistabelid protsessi teksti- ja andmesegmentide kohta
  - Erineva protsessi tekstisegmendid võivad saalimisala jagada
- Solaris, Linux — saalimisala reserveeritakse vastavalt vajadusele
  - Tekstisegmente ei saalita, vaid kasutatakse originaalfaili



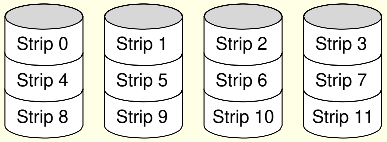


# RAID massiivid

- RAID — *Redundant Array of Inexpensive Disks*
- Paralleelsuse kaudu kiiruse suurendamine
- Töökindluse suurendamine liiasusega info salvestamisel
  - Peegeldamine (*mirroring, shadowing*)
  - Paarsusinfo laiali jagamine (*interleaved parity*)
- Parandusaeg peab mõistlik olema
- Kuumvaru (*hot spare*)
- Tarkvaraline vs riistvaraline RAID

# RAID tasemed

- RAID 0 — Mitmest kettast ühe suure tegemine (*striping*)
- Virtuaalne ketas koosneb vöötidest; igas vöödis $k$ sektorit
- Vöödid jagatud ühtlaselt $n$ ketta vahel



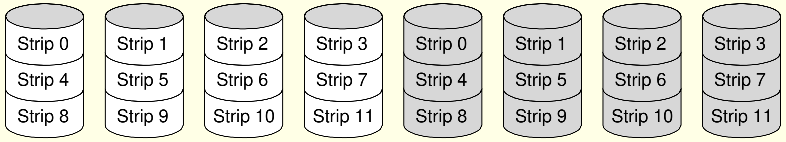

- Kõrvuti olevate vöötide lugemine võib toimuda paralleelselt
- Parandab jõudlust, kuid ei tõsta töökindlust
- Kui süsteem loeb kettalt sektorhaaval, siis pole efektiivsuses võitu





# RAID tasemed

- RAID 1 — Peegeldamine (sama info kahe seadme peal)



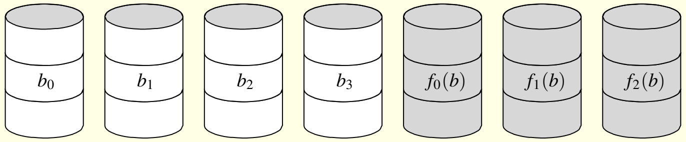

- Säilivad kõik RAID 0 eelised
- Vöödi lugemine sellelt kettalt, kust on parasjagu kiirem
- Kirjutamised peavad toimuma mõlemal kettal, kuid saab teha paralleelselt





# RAID tasemed

- RAID 2 — Mälu stiilis veaparanduskoodid (ECC)



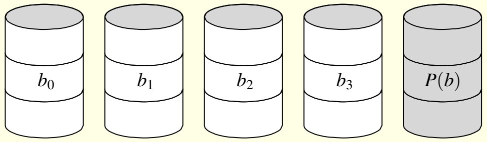

- Vöödid on väikesed (biti, baidi või sõna suurused)
- Lisaks salvestatakse Hammingu kood
- Ühe ketta veast suudab taastuda
- Vajab log lisakettaid
- Harva kasutusel





# RAID tasemed

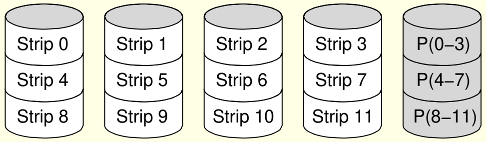
- RAID 3 — Bitihaaval paarsuskontroll




- Vöödid on väikesed (biti, baidi või sõna suurused)
- Liiasuse saavutamiseks leitakse tegelike andmete XOR
- Ühe ketta veast suudab taastuda
- Vajab üht lisaketast

# RAID tasemed

- RAID 4 — Plokihaaval paarsuskontroll



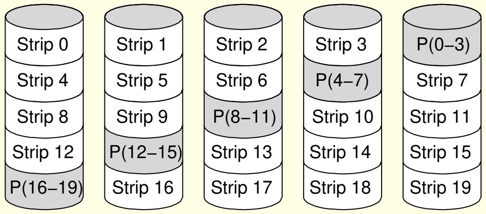

- Sarnane RAID 3ga, kuid vöödisuurus suurem
- Iga kirjutusoperatsioon nõuab paarsusinfo korrigeerimiseks kõikide ketaste lugemist
- Alternatiivselt arvutatakse uus paarsusinfo vanast andme- ja paarsusinfost (2 lisalugemist)
- Paarsusketas võib osutuda pudelikaelaks





# RAID tasemed

- RAID 5 — Plokihaaval hajutatud paarsuskontroll



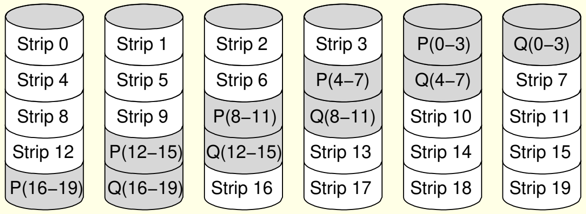

- Nagu RAID 4, aga paarsussektorid hajutatud ketaste vahel
- Väldib eraldi paarsuskettaga seotud pudelikaela





# RAID tasemed

- RAID 6 — P+Q liiasus




- Nagu RAID 5, aga liiasust arvutatakse kahe eraldi funktsiooni abil
- Vajab kaht lisaketast
- Suudab taastuda kahe ketta vea korral





# Stabiilne salvestusruum

- Salvestusruum, mis ”mittekunagi” ei ole vigane
- Realiseeritud tarkvaraliselt kasutades kahte (või rohkemat) peegeldatud ketast
  - Stabiilne kirjutamine
  - Stabiilne lugemine
  - Vigadest taastumine