UNIVERSITETET
I OSLO

# Effektiv bruk av sekundærlager

# Beregningsmodeller

- **RAM-modellen** (RAM = Random Access Model):
  - Antar at alle data får plass i primærminnet
  - Tiden det tar å aksessere et data item er uavhengig av hva innholdet i itemet er eller hvor det ligger i minnet
- **I/O-modellen**:
  - Når ikke alle data kan få plass i primærminnet samtidig
  - Antall blokkaksesser (skriving/lesing) er det som avgjør hvor god en algoritme er (Tidsbruk i primærminnet er neglisjerbar)

# Sortering

- Små mengder data (alt får plass i primærminnet):
  - RAM-modellen
  - Variant av **Quicksort**
- Store mengder data (kan ikke unngå diskaksesser):
  - I/O-modellen
  - **Two-phase, multiway merge-sort (TPMMS)**

# Merge-sort

- **Merge-sort** er en algoritme for primær-minne-sortering (RAM-modellen)
- Anta n elementer skal sorteres
  - *Basis:* En liste med ett element.
    - Listen er allerede ferdigsortert
  - *Induksjon:* Gitt en liste med mer enn ett element.
    - Del listen i to like store lister.
    - Sorter hver liste med merge-sort.
    - Flett de resulterende listene til en sortert liste.

# TPMMS I

**TPMMS – Two-Phase Multiway Merge-Sort** – er en algoritme for sortering ved store datamengder (I/O-modellen)

- Fase 1: Data grupperes i bolker som hver får plass i primærminnet.
  - Sorter hver gruppe (bruk f.eks. Quicksort) og legg tilbake på disk
- Fase 2: Flett alle sorterte sublister:
  - Sett av én inputbuffer i primærlageret pr. subliste pluss én outputbuffer, hver buffer rommer én blokk. Last opp første blokk fra hver av sublistene.
  - Finn minste element i inputbufferne, flytt det til outputbufferet (bruk f.eks. lineært søk, det tar kort tid sammenliknet med henting av blokker fra disk). Gjenta med gjenværende elementer.
    - Når en inputbuffer er tom, hentes neste blokk i tilhørende subliste fra disk til inputbuffer
    - Når outputbufferet er fullt, skrives det til disk

# TPMMS II

- Fase 1:
  - Blokkene kan leses i rekkefølge slik de ligger lagret på disk
  - Hver blokk leses fra og skrives til disk en gang
- Fase 2:
  - Blokkene leses i uforutsigbar rekkefølge fra disk
  - Hver blokk leses nøyaktig en gang
  - Hver record flyttes til outputbufferet nøyaktig en gang, og skrives til disk en gang. Totalt skrives like mange blokker som det som blir lest.

# TPMMS III

- Når ”sprekker” TPMMS? La

  B – blokkstørrelse

  M – plass avsatt i primærminnet til bufring

  R – recordstørrelse

  (alle i antall bytes)

  – Hver subliste fra fase 1 er maks M bytes lang, dvs. inneholder maks M/R records
  – Antall sublister som kan håndteres i fase 2, er maks (M/B-1) (antall bufre er M/B, en av dem må brukes til output)
  – Totalt antall records som kan sorteres, er derfor
    $(M/R)*((M/B)-1) \approx M^2/RB$

# Eksempel:

B= $2^{14}$ = 16384 bytes

M = 100 MB = 100 * $2^{20}$ bytes = 6400 blokker

R = 160 bytes

$M^2/RB$ = 4.2 * $10^9$ records = 0,67 terabytes

– stort nok til praktisk talt alle realistiske formål

# TPMMS IV

Hva hvis TPMMS likevel ikke klarer å behandle alle records i løpet av de to fasene?

1. Bruk TPMMS til å sortere grupper av $M^2/RB$ records, hver blir en sortert subliste
2. Bruk en tredje fase til å flette opp til (M/B-1) slike lister, totalt kan sorteres ca. $(M^2/RB)(M/B) = M^3/RB^2$ records

Eksempelet fra forrige foil:

$M^3/RB^2 = 27 * 10^{12}$ records = 4,3 petabytes