IN2010 uke 10

Jakob Hansen

4. november 2020

Hva vi skal gjøre idag

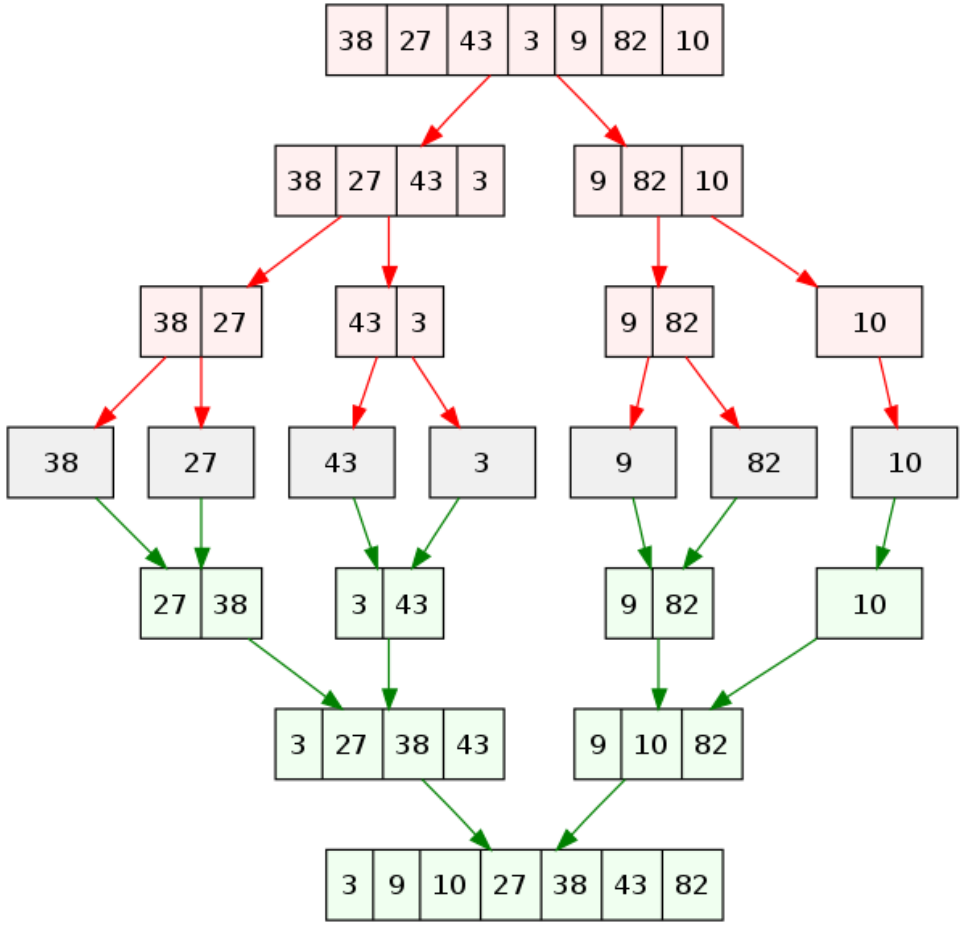
- Repetisjon?
- Mer (og vanskeligere) sortering!

En liten rettelse/oppgave fra forrige gang

- Hva var stabilitet i sortering?
- Er selection sort stabil? -> Nei!
- $[2, 3, 2, 1]$

Merge sort

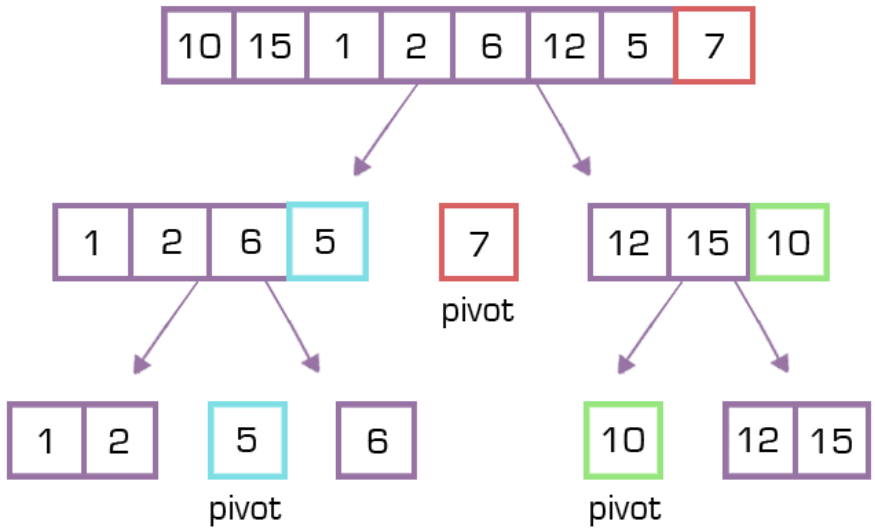
- Tankegang: Rekursivt sorter hver halvdel, sett delene sammen



- Kompleksitet? -> $O(n * log(n))$
- Stabilitet? -> Ja!
- Inplace? -> Nei, hvertfall for den vanligste implementasjon

Quick sort

- Tankegang: Velg ut et pivotelement, så nærme medianen av arrayet som mulig. Plasser alle elementer mindre enn pivot til venstre, alle større til høyre. Gjør dette rekursivt for venstre og høyre halvdel.



- Kompleksitet? -> worstcase $O(n^2)$, men average case $O(n * log(n))$
- Stabilitet? -> Nei
- Inplace? -> Ja!

Bucket sort

- Tankegang: Putt alle elementer i bøtter, sorter (om nødvendig) og sett sammen.
- Implementasjonen avhenger av problemet
- Veldig rask hvis vi har en viss informasjon om hva vi sorterer (range på nøkler for eksempel)
- Utnytter ofte andre sorteringsalgoritmer for å sortere innad i bøttene
- Ikke comparisonbasert! (hvertfall den klassiske implementasjon)
- Kompleksitet? -> Hvis vi ikke sorterer: $O(n)$, hvis vi må sortere: Kompleksiteten til sorteringsalgoritmen
- Stabilitet? Ja, sålenge sorteringsalgoritmen er stabil
- Inplace? Nei

Radix sort

- Leksikografisk sortering. Kan også beskrives som kategorisk sortering.
- For tall: sorter først på 1'er plassen, så 10'er plassen, så 100'er plassen...
- Her kan vi utnytte bucket sort!
- Siden vi sorterer etter ett og ett tall, kan vi ha 10 bøtter, 0-9
- Kompleksitet? -> O(n*d) der d er maksimalt antall siffer
- Stabilitet? Virker kanskje ikke sånn, men ja!
- Inplace? Nja, bruker jo bucket sort, så nei

En liten digresjon utenfor pensum -> Timsort

- Timsort er designet for å brukes i virkeligheten
- Er standard sorteringsalgoritmen i blant annet Java, Python, JS, Rust ...
- Analyserer arrayet for å finne ut hvilken approach som er best
- Under 64 elementer -> Insertion sort
- Over 64 elementer -> En variant av mergesort (+ binærsøk!)
- Gjør mye lurt for å bruke mindre minne, samt utnytte naturlige mønstre
- Kompleksitet? -> worst case $O(n * log(n))$, best case $O(n)$