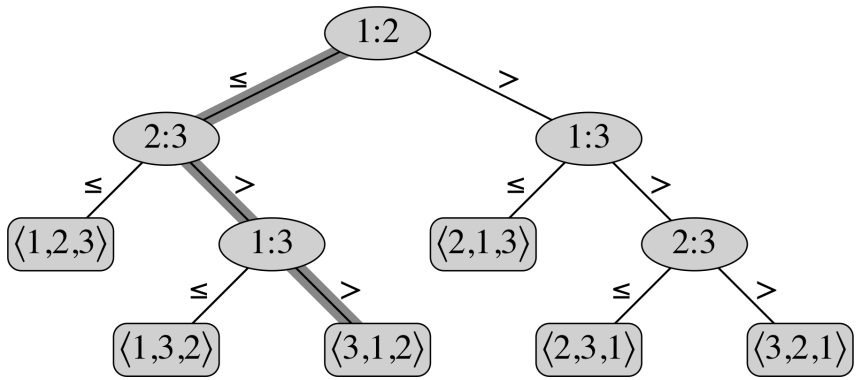
Sortering i lineær tid

Sammenligningsbaseret sortering

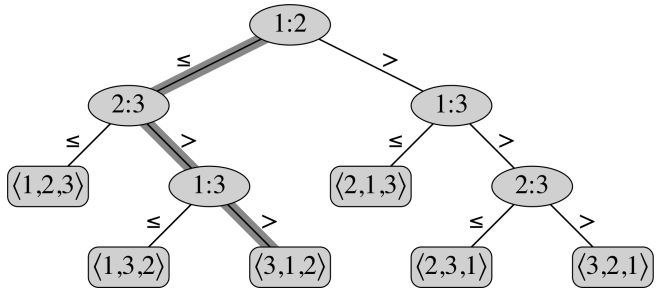
Nedre grænser kræver en præcis beregningsmodel.

Model for sammenligningsbaserede sorteringsalgoritmer:



Worst-case køretid: længste rod-blad sti = træets højde.

Sammenligningsbaseret sortering



For en fast samling af n elementer er der $n! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 \cdot \ldots \cdot n$ forskellige input (rækkefølger af elementer).

Hvis algoritmen (træet) skal kunne sortere alle disse, skal der være mindst $n!$ blade - ellers vil der være to forskellig input som leder til samme svar, og for det ene input må svaret være forkert.

Et træ af højde h har højst 2^h blade.

$$2^h \geq \text{ antal blade } \geq n!$$

$$h \geq \log(n!) = \log(1 \cdot 2 \cdot 3 \ldots \cdot n)$$

$$= \log(1) + \log(2) + \log(3) + \cdots + \log(n) \geq \frac{n}{2} \cdot \log(\frac{n}{2}) = \frac{n}{2}(\log(n) - 1)$$

$$h = \Omega(n \log n)$$

Counting sort

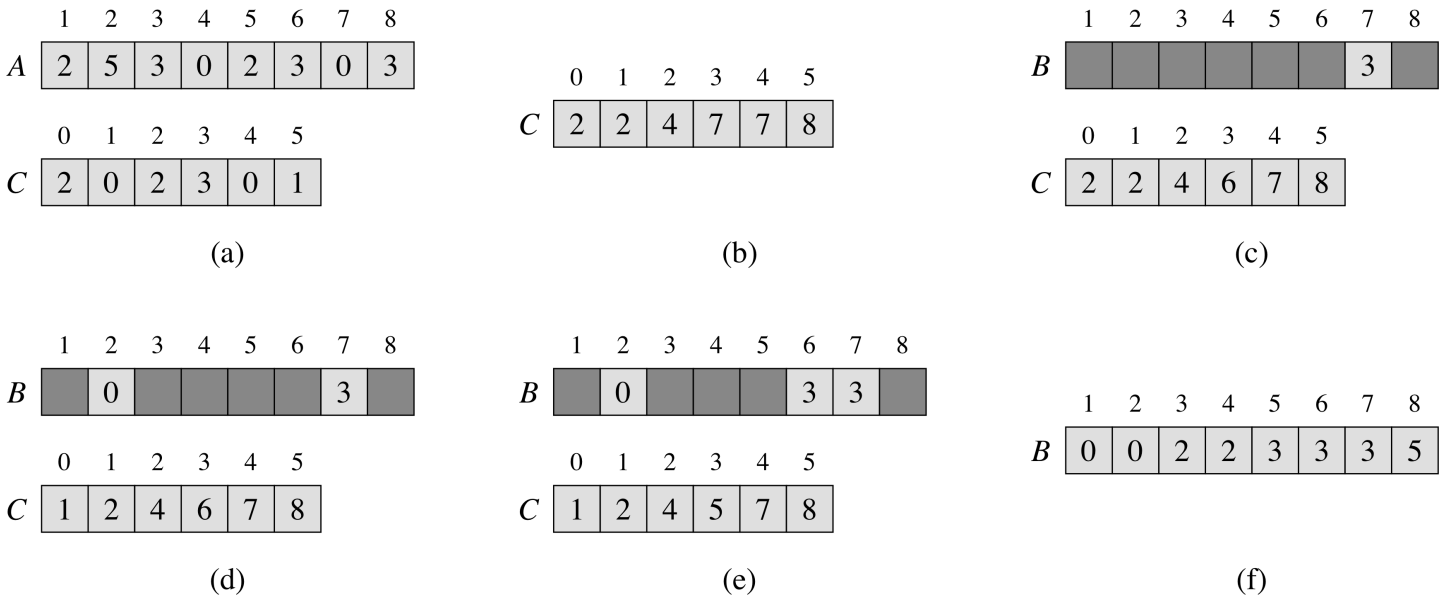
Elementer heltal: elementer kan bruges som array-indekser ($\neq$ at bruge sammenligninger på elementer).

Counting sort: Sorterer n heltal af størrelse mellem 0 og k.

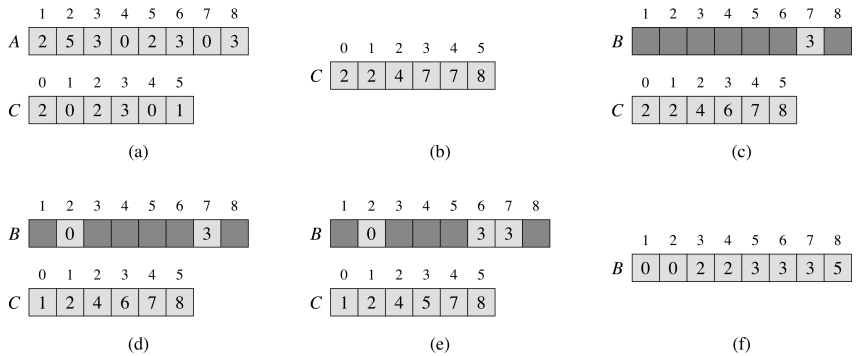
Inputarray: A (længde n)

Outputarray: B (længde n)

Array af tællere for hver mulig elementværdi: C (længde $k+1$)



Counting sort



COUNTING-SORT(A,B,k)

for $i = 0$ **to** k
  $C[i] = 0$
for $j = 1$ **to** $A.length$
  $C[A[j]] + +$
for $i = 1$ **to** k
  $C[i] = C[i] + C[i-1]$
for $j = A.length$ **downto** 1
  $B[C[A[j]]] = A[j]$
  $C[A[j]] - -$

Tid: $O(n+k)$

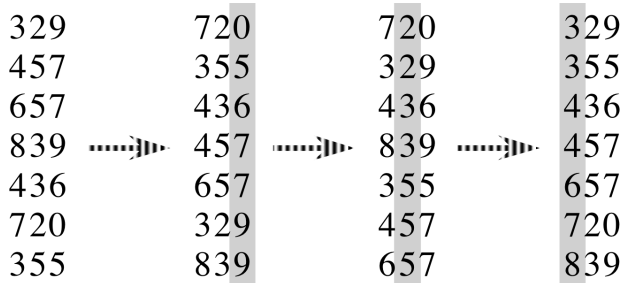
Bemærk: stabil (da sidste løkke løber baglæns gennem både A og B), dvs at elementer med ens værdier beholder deres indbyrdes plads.

Radix sort

Radix sort: Sorterer n heltal alle med d cifre i base (radix) k.

(dvs. cifrene er heltal i $\{0, 1, 2, \ldots, k-1\}$)

RADIX-SORT(A,d)
for $i = 1$ **to** d
use a stable sort to sort A on digit i from right



Tid: $O(d(n+k))$ hvis der bruges Counting Sort i løkken.

Korrekthed via løkkeinvariant:

Efter i'te iteration er løkken er A sorteret hvis man kun kigger på de i cifre mest til højre.

Radix sort

Eksempel: 32-bits heltal.

Se som 2-cifrede tal i base 2^{16}.

Radixsort sorterer disse i tid $O(2(n + 2^{16}))$

Dette er $O(n)$ hvis $n \geq 2^{16} = 65.536$

Eller se som 4-cifrede tal i base 2^8.

Radixsort sorterer disse i tid $O(4(n + 2^8))$

Dette er $O(n)$ hvis $n \geq 2^8 = 256$