



Stof

- slides college 6
- boek 4.1, 4.3

Opgaven

1. Geef een best-case en een worst-case input-rijtje voor insertion sort.
2. Is insertion sort een in-place sorteeralgoritme?
3. Geef pseudo-code voor insertion sort dat tot doel heeft het sorteren van de input-array $A[0 \dots (n-1)]$ maar dan aflopend (\geq) in plaats van oplopend. Hoe kun je met een invariant de correctheid van je algoritme aantonen?
4. Geef insertion sort als een recursieve procedure: om $A[0 \dots n]$ te sorteren, sorteren we $A[0 \dots n-1]$, en voegen we dan $A[n]$ op de juiste plek toe. Geef een functie voor de tijdscomplexiteit van de recursieve insertion sort, en laat daarmee zien wat de worst-case tijdscomplexiteit in termen van grote-Oh is.
5. Een *inversie* in een rij getallen is een tweetal i en j uit die rij zodat i vóór j komt en $i > j$.
 - (a) Geef de inversies van $[2, 3, 8, 6, 1]$.
 - (b) Geef een permutatie van $[1, \dots, 10]$ met zoveel mogelijk inversies.
 - (c) Wat kun je zeggen over insertion sort en de inversies in het input-rijtje?
6. (Dit is ongeveer opgave C-4.19.) Beschrijf (hoeft niet in pseudo-code) een algoritme in $\mathcal{O}(n \log n)$ dat als input een rij neemt en als output geeft het aantal inversies van die rij. Wat is een alternatief in $\mathcal{O}(n^2)$?
7. Is de $n-2$ in de eerste for-loop in de pseudo-code voor selection sort correct? Waarom (niet)?
8. Geef een best-case en een worst-case input-rijtje voor selection sort.
9. Toon (met een invariant) de correctheid van selection sort aan.

10. Geef (met korte uitleg) de worst-case tijdscomplexiteit van selection sort.
11. Geef de merge-sort-boom voor het sorteren van de rij $[0, 9, 7, 1, 4, 2, 5, 3, 6, 8]$.
12. Stel dat we het algoritme `merge` uit `mergeSort` gebruiken om twee gesorteerde lijsten te mergen, waarbij de eerste lengte 10 en de tweede lengte 20 heeft. Wat is het minimum aantal vergelijkingen dat nodig is? En wat het maximum?
13. De recurrente betrekking die bij merge-sort hoort is:

$$T(n) = \begin{cases} 1 & \text{als } n = 1 \\ 2T(\frac{n}{2}) + n & \text{als } n > 1 \end{cases}$$

Los deze recurrente betrekking op voor n een macht van 2, en geef aan wat dus de tijdscomplexiteit van merge-sort is in termen van grote- \mathcal{O} .

14. Beargumenteer dat de asymptotische afchatting in de vorige opgave ook goed is voor n niet een 2-macht.
15. Geef een recursief algoritme voor het vinden van een maximum element in een array van n elementen. Wat is de tijdscomplexiteit van het algoritme?
16. Geef de quicksort-boom voor het sorteren van de rij $[8, 5, 2, 7, 1, 3, 4, 6, 9]$; gebruik als pivot steeds het laatste element.
17. We bekijken quicksort waarbij als pivot steeds het middelste element (of dat net links van het midden, bij een even aantal) van de input-rij wordt gekozen. Geef een voorbeeld (bijvoorbeeld van lengte 7) waaruit blijkt dat de worst-case tijdscomplexiteit in $\mathcal{O}(n^2)$ is.
18. Geef een recursief algoritme dat als input neemt een array van n elementen, en indexen $0 \leq i < j \leq n - 1$, en de het subarray van index i tot en met j omdraait. Pas je algoritme toe op het array $[4, 3, 6, 2, 5]$ en indexen $i = 0$ en $j = 4$.
19. We bekijken als input een array A van n bits. Wat is de tijdscomplexiteit voor het sorteren van A met merge-sort? En met quick-sort?
20. We bekijken quick-sort met pivot $\lfloor n/2 \rfloor$. Wat is de running time voor een input-rijtje dat al gesorteerd is?
21. Geef een voorbeeld van een input-rijtje dat running time in $\mathcal{O}(n \log(n))$ heeft voor merge-sort en voor heap-sort, maar in $\mathcal{O}(n)$ voor insertion-sort.
22. Geef een algoritme in $\mathcal{O}(n \log n)$ dat als input een rij A van n gehele getallen neemt, en als output levert een rij waarin elk getal uit A precies één keer voorkomt (dus de dubbele worden uit A weggehaald).