Complexiteit

Uitwerking opgave 23+24+25

Opgave 23

a. Een algoritme dat de mediaan van a, b en c bepaalt:

```
        if a < b then
(I1)        if b < c then m := b
(I2)        else if a < c then m := c
(I3)                else m := a
                    fi
            fi
(I4)    else if a < c then m := a
(I5)        else if b < c then m := c
(I6)                else m := b
                    fi
            fi
        fi
```

Er zijn overigens ook geheel andere oplossingen, bijvoorbeeld: reken het gemiddelde van de drie getallen uit; diegene van de drie die het dichtst bij dit gemiddelde ligt is de mediaan. Er zijn dan geen onderlinge vergelijkingen tussen de getallen a, b en c nodig, maar wel andere operaties! Let er dus goed op dat je beweringen over het juiste model gaan, bijvoorbeeld —meestal— over aantallen vergelijkingen tussen sleutels.

b. De mogelijke invoeren zijn dus $I_1 : a < b < c$, met 2 vergelijkingen, $I_2 : a < c < b$ met 3, $I_3 : c < a < b$ met 3, $I_4 : b < a < c$ met 2, $I_5 : b < c < a$ met 3 en $I_6 : c < b < a$ met 3 vergelijkingen. Het aantal vergelijkingen noteren we als $t(I_j)$.

c. In het slechtste geval (I_2, I_3, I_5 en I_6) worden 3 vergelijkingen gedaan, in het beste geval (I_1 en I_4) 2 stuks. Voor het gemiddelde geval nemen we aan dat alle invoertypen een even grote kans hebben, dus $p(I_j) = 1/6$ voor alle $1 \leq j \leq 6$. Dan is het gemiddelde aantal vergelijkingen gelijk aan

$$\sum_{j=1}^{6} p(I_j)t(I_j) = \frac{1}{6}(2 + 3 + 3 + 2 + 3 + 3) = 8/3.$$

d. Merk op: voor $n = 3$ zijn er in essentie $3! = 6$ mogelijke te onderscheiden invoerrijtjes, namelijk I_1 t/m I_6.

We laten zien dat 3 vergelijkingen voor het vinden van de mediaan noodzakelijk zijn in de worst case. We bewijzen dat met behulp van een adversary-argument.

Zonder beperking der algemeenheid mogen we aannemen dat het algoritme begint met het vergelijken van a en b. De adversary antwoordt hierop dat $a < b$. De enige mogelijke invoerrijtjes I die daarmee corresponderen zijn I_1, I_2 en I_3.

- Stel dat het algoritme als tweede vergelijking b met c vergelijkt Als $b < c$, dan voldoet daaraan alleen het rijtje $I = I_1$ en zou de mediaan bekend zijn. De adversary antwoordt dus dat $c < b$. In dat geval weet het algoritme nog niet of $I = I_2$ of $I = I_3$ en is dus ook de mediaan onbekend. Er is dan zeker nog een vergelijking nodig.

- Stel dat het algoritme als tweede vergelijking a met c vergelijkt. De adversary antwoordt daarop dat $a < c$. Het algoritme weet nu nog niet of $I = I_1$ of $I = I_2$ en derhalve ook nog niet of b of c de mediaan is. Er is dus nog minstens één vergelijking nodig.
- Stel het algoritme vergelijkt als tweede vergelijking a met b. Dit is een nutteloze vergelijking die geen extra informatie geeft. Het algoritme doet er dan allen maar nog langer over om de mediaan te vinden.

We zagen dus: 2 vergelijkingen zijn tegen de adversary niet voldoende. Er is altijd een bad case waarop het algoritme ten minste drie vergelijkingen nodig heeft. Derhalve is het aantal vergelijkingen in het ergste geval altijd ten minste 3.
Bij onderdeel **a.** zagen we dat het met 3 vergelijkingen wel kan, dus is 3 noodzakelijk en voldoende. Conclusie: het algoritme van **a.** is optimaal.

Opgave 24
Vergelijk twee elementen met elkaar, en ook nog eens twee andere. Vergelijk de "winnaars". Na drie vergelijkingen en na eventueel hernoemen van de variabelen weten we dan —zoals gebruikelijk— dat $a < b < d$ en $c < d$. Dus d is een van de twee grootste elementen en kan de mediaan niet zijn. Resteert het vinden van de op een na grootste van $\{a, b, c, e\}$ met $a < b$. Vergelijk c en e. Neem aan dat $c < e$ (verwissel anders de rol van c en e). Vergelijk b en e, en na vijf vergelijkingen hebben we —na eventueel hernoemen van de variabelen— $a < b < e$ en $c < e$. Vergelijk nu b en c, en de winnaar is de mediaan die we zochten: zes vergelijkingen.
Merk op dat we bijvoorbeeld niet weten wie van d en e de grootste is! Het sorteren van vijf elementen is essentieel moeilijker dan het vinden van de mediaan van die vijf elementen. Op college hebben we gezien dat de mediaan —en meer algemeen de k-de in grootte— in lineaire tijd, dus $O(n)$, gevonden kan worden, terwijl sorteren minstens $\Theta(n \lg n)$ array-vergelijkingen vergt.

Opgave 25
a. Zorg ervoor dat je na 3 vergelijkingen in een situatie bent waarin je van a, b, c en d weet dat $a < b < d$ en $c < d$ (ga na dat dit mogelijk is, na eventueel hernoemen). Vergelijk nu b en c, en eventueel nog a en c. Klaar. Bijvoorbeeld binary insertion sort (zie college/dictaat over optimaal sorteren) doet iets vergelijkbaars.
b. We maken gebruik van het algoritme van Demuth uit 1956, dat met maximaal 7 vergelijkingen 5 getallen sorteert. (Vergelijk weer binary insertion sort.) De methode gaat als volgt. Vergelijk het eerste met het tweede getal, en het derde met het vierde; vergelijk nu de twee grootste. Je hebt dan weer $a < b < d$ en $c < d$, en e is nog over; er zijn 3 vergelijkingen gedaan. Via binair zoeken kunnen we met 2 vergelijkingen de plek vinden waar e in de keten $a < b < d$ moet staan: dan resteren vier situaties, namelijk $e < a < b < d$, $a < e < b < d$, $a < b < e < d$ en $a < b < d < e$. Stop nu c in de ontstane rij van 4 getallen, waarbij we weten dat $c < d$ is. Dat kost (maximaal) 2 vergelijkingen via binair zoeken, 1 minder dan wanneer je $c < d$ niet zou weten. Je zoekt nu immers de positie van c in een gesorteerd rijtje van maximaal 3 getallen, in plaats van 4. Wat je eigenlijk doet is de voor binair zoeken ongunstige situatie (zoeken in 4 elementen) "gratis" ombuigen naar een situatie van zoeken in 3 elementen omdat je al weet dat $c < d$.