Hertentamen Complexiteit
Maandag 10 juli 2023, 9.00 – 12.00

Geef een *duidelijke* toelichting/uitleg bij al je antwoorden!

Opgave 1. (25 punten)

Gegeven is een array $A = A[1], \ldots, A[n]$ dat n verschillende gehele getallen bevat, met $n = 3k$, n oneven en $n \geq 3$. Gegeven is verder dat de eerste $\frac{n}{3}$ getallen positief zijn en de laatste $\frac{n}{3}$ getallen negatief. Van de middelste $\frac{n}{3}$ getallen is alleen bekend dat het merendeel positief is.
Voorbeeld: $9, 1, 4, 3, -1, 2, -3, -17, -7$ $(n = 9)$

We willen van dit array de index vinden van de mediaan (de middelste in grootte). In het voorbeeld is dit 2, van $A[2] = 1$.

a. (4 punten)
Gegeven een beslissingsboom corresponderend met een algoritme dat gebaseerd is op arrayvergelijkingen. In termen van het algoritme, wat stelt een pad van de wortel naar een blad voor, wat is de betekenis van de lengte van zo'n pad, wat kun je zeggen over de bladeren en wat stelt de hoogte van zo'n boom voor?

b. (10 punten)
Bewijs met behulp van een *beslissingsboomargument* dat elk algoritme voor bovenstaand probleem dat is gebaseerd op arrayvergelijkingen, voor de bekeken soort invoerrijtjes ten minste $\lceil \lg \frac{n}{3} \rceil + 1$ vergelijkingen moet doen in de worst case. Formuleer je bewijs zorgvuldig en geef aan welke stellingen/eigenschappen je gebruikt.

c. (3 punten)
In het college hebben we een algoritme gezien dat de mediaan kan bepalen in $O(n)$. Kunnen we op basis van de ondergrens uit **b** concluderen dat dat algoritme niet optimaal is?

d. (5 punten)
Hoeveel vergelijkingen doet Insertion sort in de worst case voor het sorteren van de bekeken soort invoerrijtjes?

e. (3 punten)
Shellsort met de Hibbard-stapgroottes doet in de worst case $\Theta(n\sqrt{n})$ vergelijkingen. Dit is een ordeverbetering op Insertion sort. Toch is Insertion sort optimaal bewezen binnen een bepaalde klasse van algoritmen. Om welke klasse van algoritmen gaat dit, en waarom voldoet Shellsort hier niet aan?

Opgave 2. (15 punten)

Gegeven is een volledige[1] graaf $\mathcal{G}$ met n knopen (n is even en $n \geq 2$) en met gewichten op de takken. Hierin construeren we recursief een pad door de graaf dat alle knopen precies één keer bezoekt, waarbij we een beetje letten op het totale gewicht. Voor $n \geq 4$ kiezen we een willekeurige knoop p en zoeken we de bijbehorende knoop q zodat het gewicht van de tak tussen p en q zo laag mogelijk is. We construeren *recursief* een pad in de rest van de graaf, d.w.z., de graaf zonder p en q en de bijbehorende takken. Vervolgens plakken we de tak tussen p en q voor- of achteraan het recursief geconstrueerde pad op de manier die het laagste gewicht geeft. Voor $n = 2$, het basisgeval, nemen we direct de (enige) tak die er is. Het aantal keren dat we het gewicht van een tak opvragen noemen we $W(n)$.

[1] Een graaf is *volledig* als *elk* paar knopen is verbonden door een tak.

a. (5 punten)
Leg uit waarom $W(n)$ voldoet aan de volgende recurrente betrekking:

$$W(n) = \begin{cases} 0 & n = 2 \\ W(n-2) + n + 3 & n \geq 4, n = 2k \end{cases}$$

b. (10 punten)
Los de recurrente betrekking exact op door deze herhaald in zichzelf in te vullen (substitutiemethode). Schrijf $W(n)$ als functie van n. Bewijs vervolgens met behulp van volledige inductie dat de aldus gevonden oplossing inderdaad voldoet. Je mag hierbij gebruiken dat:

$$\sum_{i\,=\,1,\ i\ \text{oneven}}^{\ell} i = \frac{\ell^2 + 2\ell + 1}{4}$$

Opgave 3. (25 punten)

Gegeven is een array $A = A[1], \ldots, A[n]$ met $n \geq 2$ positieve gehele getallen. Een element heeft *overwicht* k als het groter is dan de *som* van de volgende k elementen: $A[i] > A[i+1] + \ldots + A[i+k]$ (waar $i+k \leq n$). Hierbij moet k maximaal zijn, wat wil zeggen dat $A[i] \leq A[i+1] + \ldots + A[i+k+1]$ of dat $i + k = n$.

Voorbeeld: in de rij $8, 10, 2, 6, 1, 9, 1, 4$ hebben de elementen respectievelijk overwicht $0, 3, 0, 1, 0, 2, 0, 0$.

We bekijken een algoritme dat de index oplevert van het element met het *meeste overwicht*. In het voorbeeld is dit 2 (van $A[2] = 10$), met overwicht 3. Als er meerdere elementen zijn met het meeste overwicht, wordt de index van het meest linker van deze elementen geretourneerd.
Op regel 4 geldt dat, als de eerste test onwaar is, de tweede test niet wordt uitgevoerd.

```
1  i := 1; meest_index := 0; meest_overwicht := −1;
2  while i < n do
3  |  j := i + 1; som := A[j];
4  |  while j ≤ n and A[i] > som do
5  |  |  j := j + 1;
6  |  |  if j ≤ n then
7  |  |  |  som := som + A[j];
8  |  |  fi
9  |  od
10 |  if j − 1 − i > meest_overwicht then
11 |  |  meest_index := i;
12 |  |  meest_overwicht := j − 1 − i;
13 |  fi
14 |  i := i + 1;
15 od
16 return meest_index;
```

a. (5 punten)
Laat zien dat de vergelijking $A[i] > som$ op regel 4 maatgevend is voor de complexiteit van het algoritme; m.a.w., laat zien dat *alle* andere operaties in het algoritme (waaronder de vergelijking $j \leq n$ op regel 4) in orde van grootte hooguit even vaak gebeuren als $A[i] > som$.

b. (10 punten)
Hoeveel vergelijkingen $A[i] > som$ doet dit algoritme in de *best case*? Geef een exact antwoord en beschijf *alle* bijbehorende invoerrijtjes. Leg op basis van het algoritme uit hoe je aan je antwoord komt.

c. (10 punten)
Hoeveel vergelijkingen $A[i] > som$ doet dit algoritme in de *worst case*? Geef een exact antwoord en beschrijf *alle* bijbehorende invoerrijtjes. Leg op basis van het algoritme uit hoe je aan je antwoord komt.

Opgave 4. (35 punten)

We bekijken de volgende twee beslissingsproblemen:

> **Subset Sum (SUM)**: gegeven een getal $D \in \mathbb{N}$ en een eindige verzameling $S \subseteq \mathbb{N}$.
> **Vraag**: bestaat er een deelverzameling van S zodat de elementen ervan sommeren tot D, m.a.w., bestaat er een $S' \subseteq S$ zodat $\sum_{s \in S'} s = D$?
>
> **Partitie**: gegeven een (eindig) rijtje getallen $(a_1, \ldots, a_n) \in \mathbb{N}^n$ (m.a.w., $a_i \in \mathbb{N}$).
> **Vraag**: zijn de getallen op te delen in twee deelrijtjes zodat de som van beide deelrijtjes hetzelfde is, m.a.w., bestaat er een deelverzameling $I \subseteq \{1, \ldots, n\}$ zodat $\sum_{i \in I} a_i = \sum_{i \notin I} a_i$?

Voorbeeld (Partitie): het rijtje $(3, 1, 1, 2, 2, 1)$ is op te delen in deelrijtjes $(3, 2)$ en $(1, 1, 2, 1)$, die beide sommeren tot 5.

a. (10 punten)
Toon aan dat Partitie $\in \mathcal{NP}$ door een *niet-deterministisch polynomiaal* algoritme te geven voor Partitie. Het algoritme heeft dus als invoer een rijtje getallen $x = (a_1, \ldots, a_n) \in \mathbb{N}^n$. Het algoritme moet "ja" opleveren dan en slechts dan als x een ja-instantie is voor Partitie (&).
Leg uit wat in elke fase van het algoritme gebeurt *en hoe*, en hoeveel stappen dat kost. Leg ook uit waarom jouw algoritme voldoet aan (&) en waarom het polynomiaal is in $|x|$.

We bekijken een transformatie T die instanties van SUM transformeert naar instanties van Partitie. Gegeven een getal $D \in \mathbb{N}$ en een eindige verzameling $S = \{s_1, \ldots, s_n\} \subseteq \mathbb{N}$. We definiëren de transformatie T als: $T(\langle D, S \rangle) = (s_1, \ldots, s_n, t_1, t_2)$, waar $t_1 = \sum_{s \in S} s$ en $t_2 = 2D$. Met andere woorden: we nemen alle elementen van S en we voegen er twee getallen aan toe: de som van alle elementen in S, en $2D$.

Voorbeeld: als $D = 16$ en $S = \{3, 1, 4, 5, 9, 2, 6\}$, wordt $T(\langle D, S \rangle) = (3, 1, 4, 5, 9, 2, 6, 30, 32)$.

b. (5 punten)
Toon aan: $\langle D, S \rangle$ is een ja-instantie van SUM $\implies T(\langle D, S \rangle)$ is een ja-instantie van Partitie.

c. (5 punten)
Toon aan: $T(\langle D, S \rangle)$ is een ja-instantie van Partitie $\implies \langle D, S \rangle$ is een ja-instantie van SUM.

Het is duidelijk dat de constructie van $T(\langle D, S \rangle)$ polynomiaal is. Samen met **b** en **c** volgt dan dat SUM $\leq_{\mathsf{P}}$ Partitie.

d. (2 punten)
Wanneer is een beslissingsprobleem P NP-hard? En wanneer NP-volledig? (De definitie van $\mathcal{NP}$ hoeft niet te worden gegeven.)

We hebben hierboven gezien dat (1) Partitie $\in \mathcal{NP}$ en dat (2) SUM $\leq_{\mathsf{P}}$ Partitie. Van college is bekend dat (3) SUM $\in \mathcal{NPC}$. Gegeven zijn verder een of ander probleem $Q \in \mathcal{P}$ en een of ander probleem $R \in \mathcal{EXP}$.

e. (13 punten)

(i) Welke van (1), (2) en (3) heb je nodig om te concluderen dat Partitie $\leq_{\mathsf{P}}$ SUM? Leg uit.

(ii) Welke van (1), (2), (3) en (i) heb je nodig om te concluderen dat Partitie $\in \mathcal{NPC}$? Leg uit.

(iii) Geef van de volgende vier reducties aan of ze wel, misschien of niet mogelijk zijn. Leg uit.

$$Q \leq_{\mathsf{P}} \text{SUM} \qquad \text{SUM} \leq_{\mathsf{P}} Q \qquad R \leq_{\mathsf{P}} \text{SUM} \qquad \text{SUM} \leq_{\mathsf{P}} R$$

EINDE