Tentamen Complexiteit
Maandag 15 juli 2024, 9.00 – 12.00 uur

Geef een *duidelijke* toelichting bij al je antwoorden! Veel succes!

Opgave 1. (20 punten)
Gegeven een array $A = A[1], A[2], \ldots, A[n]$ dat n verschillende getallen bevat ($n = 2^k$ voor zekere $k \geq 3$). Het array is reeds enigszins voorgesorteerd en bestaat uit $\frac{n}{4}$ opeenvolgende deelarrays van elk 4 elementen waarvoor geldt: de elementen in het j-de deelarray zijn allemaal kleiner dan alle elementen in het $j+1$-de deelarray. We willen het array oplopend sorteren.

Een voorbeeld met $n = 16$: $\underline{5,4,3,9}, \underline{14,17,15,11}, \underline{20,22,18,25}, \underline{31,28,27,33}$

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. (8 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 $\frac{3n}{4} + \lceil \frac{n}{4} \cdot \lg 3 \rceil$ arrayvergelijkingen moet doen in de worst case. Formuleer je bewijs zorgvuldig, maak gebruik van **a** en geef aan welke stellingen/eigenschappen je gebruikt.
Ter herinnering: $\lg x^y = y \cdot \lg x$ en $\lg(x \cdot y) = \lg x + \lg y$.
c. (6 punten)
Bepaal het maximale aantal inversies van A en geef aan voor wat voor rijtjes dit aantal gehaald wordt. Gebruik dat aantal om te berekenen hoeveel arrayvergelijkingen Insertion sort of Bubble sort ten minste moet doen in de worst case op het array A. Geef duidelijk aan welke stelling(en)/eigenschappen je gebruikt (ten minste twee).
d. (2 punten)
Shellsort gebruikt Insertion sort voor het sorteren van deelrijtjes. Geldt de in **c** afgeleide ondergrens dus ook voor Shellsort? Leg je antwoord uit.

Opgave 2. (25 punten)
Gegeven is een array $A = A[1], A[2], \ldots, A[n]$ dat verschillende getallen bevat en waarbij n even is. We gaan het array recursief oplopend sorteren op de volgende manier. Eerst bepalen we de grootste en de op-een-na-grootste waarde, die we vervolgens in de juiste volgorde achteraan zetten door ze te verwisselen met de twee achterste elementen. Daarna sorteren we de rest van het array op dezelfde manier.

a. (10 punten)
We zouden de toernooimethode kunnen gebruiken om de grootste en de op-een-na-grootste waarde te bepalen. Leg uit hoe deze methode werkt en hoeveel arrayvergelijkingen deze doet. Vergeet daarbij niet uit te leggen hoe de op-een-na-grootste wordt bepaald. Geef ook aan

waarom de methode het correcte antwoord geeft. Je mag aannemen dat $n = 2^k$, met $k \geq 1$. Illustreer je uitleg ten slotte met het rijtje $3, 14, 6, 12, 9, 16, 4, 15, 8, 10, 13, 2, 5, 1, 11, 7$.
b. (5 punten)
We gebruiken de toernooimethode toch maar niet. In plaats daarvan bepalen we de grootste en de op-een-na-grootste waarde door telkens de twee volgende array-elementen te bekijken, en daarmee de tot dusver gevonden grootste en op-een-na-grootste te updaten. Laat zien dat deze methode met $\frac{3}{2}n - 2$ arrayvergelijkingen kan.

Laat $T(n)$ het aantal arrayvergelijkingen zijn dat ons recursieve sorteeralgoritme doet. Dan wordt $T(n)$ gegeven door de volgende recurrente betrekking:

$$T(n) = \begin{cases} 1 & n = 2 \\ T(n-2) + \frac{3}{2}n - 2 & n > 2, n = 2k \end{cases}$$

c. (10 punten)
Los de recurrente betrekking exact op door deze herhaald in zichzelf in te vullen (substitutiemethode). Schrijf $T(n)$ als functie van n. Bewijs vervolgens met behulp van volledige inductie dat de aldus gevonden oplossing inderdaad voldoet.
Hint: neem bij je uitwerking de factoren $\frac{3}{2}n$ apart en idem de factoren 2.
Verder is $\sum_{i=1}^{\ell-1} a \cdot i = a \cdot \sum_{i=1}^{\ell-1} i = a \cdot \frac{1}{2}\ell(\ell-1)$.

Opgave 3. (25 punten)
Gegeven is een array $A = A[1], \ldots, A[n]$ dat n gehele getallen bevat, met $n \geq 2$ en even. Het array bestaat uit twee even lange (vervlochten) deelrijtjes: op de oneven posities $(1, 3, \ldots, n-1)$ staan positieve (> 0) getallen, op de even posities $(2, 4, \ldots, n)$ staan negatieve (< 0) getallen. Beide deelrijtjes zijn oplopend gesorteerd. We bekijken een algoritme dat het aantal paren (i, j) telt waarvoor $i < j$ en $A[i] + A[j] = 0$.

Voorbeeld: de rij $1, -7, 2, -6, 2, -4, 3, -4, 5, -2, 6, -2$ ($n = 12$), bestaande uit de (oplopende) deelrijtjes $1, 2, 2, 4, 5, 6$ en $-7, -6, -4, -4, -2, -2$, heeft vijf van zulke paren: $(3, 10)$, $(3, 12)$, $(4, 11)$, $(5, 10)$, $(5, 12)$.

Op regel 4 geldt dat als de eerste test onwaar is, de tweede test niet gebeurt.

```
1  i := 1; paren := 0;
2  while i < n do
3  |  j := i + 1;
4  |  while j ≤ n and A[i] + A[j] ≤ 0 do
5  |  |  if A[i] + A[j] = 0 then
6  |  |  |  paren := paren + 1;
7  |  |  fi
8  |  |  j := j + 2;
9  |  od
10 |  i := i + 1;
11 od
12 return paren;
```

a. (5 punten)
Laat zien dat de vergelijking $A[i] + A[j] \leq 0$ op regel 4 maatgevend is voor de complexiteit van het algoritme. Breng hiertoe *alle* regels van het algoritme in verstandig verband met deze operatie, waaronder ook de vergelijking $j \leq n$ op regel 4.
b. (5 punten)
Leg uit waarom dit algoritme het juiste antwoord oplevert voor arrays van het bekeken type.
c. (7 punten)
(i) Hoe vaak doet dit algoritme de arrayvergelijking op regel 4 in de *best case*? Geef een exact antwoord en beschrijf *alle* bijbehorende invoerrijtjes. Leg op basis van het algoritme uit hoe je aan je antwoord komt.
(ii) Verwoord je beschrijving van de rijtjes ook zonder "voor alle $i \ldots$".
c. (8 punten)
(i) Hoe vaak doet dit algoritme de arrayvergelijking op regel 4 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.
(ii) Verwoord je beschrijving van de rijtjes ook zonder "voor alle $i \ldots$".

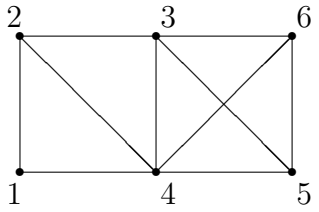
Opgave 4. (30 punten)
We bekijken de volgende twee beslissingsproblemen:

3Kleur: Gegeven een ongerichte graaf $\mathcal{G} = (V, E)$[1].
Vraag: kunnen de knopen van $\mathcal{G}$ zo gekleurd worden met hooguit drie kleuren dat geen tweetal aangrenzende knopen dezelfde kleur heeft?

CliqueCover: Gegeven een ongerichte graaf $\mathcal{G} = (V, E)$ en een geheel getal $1 < k \leq |V|$.
Vraag: bestaat er een partitie[2] van de knopen bestaande uit $\leq k$ deelverzamelingen, zodanig dat elke deelverzameling een kliek vormt in $\mathcal{G}$? Zo'n partitie zullen we een kliek-overdekking noemen.

Definitie. Een *kliek* in een ongerichte graaf $\mathcal{G} = (V, E)$ is een deelverzameling $V' \subseteq V$ zodanig dat voor elk tweetal knopen $u, v \in V'$ $(u \neq v)$ geldt dat $(u, v) \in E$. Met andere woorden: tussen elk tweetal knopen uit V' zit een tak.

Voorbeeld. Nevenstaande graaf is een ja-instantie voor CliqueCover met $k = 2$. Immers: $V_1 = \{1, 2, 4\}$ en $V_2 = \{3, 5, 6\}$ vormen samen een partitie van V en ze zijn allebei een kliek. Overigens is ook $V_1 = \{1, 2\}$ en $V_2 = \{3, 4, 5, 6\}$ een kliek-overdekking ter grootte 2.



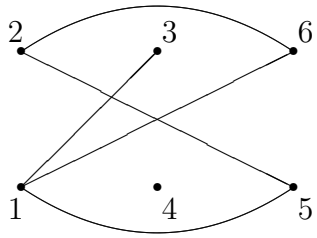
a. (8 punten)
Toon aan dat CliqueCover $\in \mathcal{NP}$ door een *niet-deterministisch polynomiaal* algoritme voor CliqueCover te geven. Het algoritme heeft dus als invoer een ongerichte graaf $\mathcal{G} = (V, E)$ en een geheel getal $k > 1$, en moet "ja" opleveren dan en slechts dan als $x = \langle \mathcal{G}, k \rangle$ een ja-instantie is voor CliqueCover (&).

[1] V en E stellen respectievelijk de knopen en de takken van $\mathcal{G}$ voor
[2] $\{V_1, V_2, \ldots, V_\ell\}$ is een partitie van V als (1) $V_i \cap V_j = \emptyset$ voor alle $i < j$, en (2) $V_1 \cup V_2 \cup \cdots \cup V_\ell = V$

Je mag aannemen dat het aantal knopen van V bekend is en gelijk is aan n.
Leg uit wat in elke fase van het algoritme gebeurt en hoeveel stappen dat kost. Leg ook uit waarom jouw algoritme aan (&) voldoet en waarom het polynomiaal is in $|x|$.

De transformatie T beeldt instanties van 3Kleur af op instanties van CliqueCover als volgt. De graaf $\mathcal{G}$ wordt afgebeeld op zijn complement $\overline{\mathcal{G}} = (V, \overline{E})$, met $\overline{E} = \{(u, v) | (u, v) \notin E\}$. Verder nemen we $k = 3$. Dus $T(\mathcal{G}) = \langle \overline{\mathcal{G}}, 3 \rangle$.

De ongerichte graaf hiernaast stelt het complement voor van de voorbeeldgraaf.



Het is duidelijk dat de constructie van $T(\mathcal{G})$ uit $\mathcal{G}$ polynomiaal is. Samen met **b** volgt dan dat 3Kleur $\leq_{\mathrm{P}}$ CliqueCover.

b. (8 punten)
Toon aan: $\mathcal{G}$ is een ja-instantie van 3Kleur $\iff$ $T(\mathcal{G})$ is een ja-instantie van CliqueCover. Formuleer daartoe eerst wat het betekent als $\mathcal{G}$ resp. $T(\mathcal{G})$ een ja-instantie is.
c. (2 punten)
Wanneer is een beslissingsprobleem Q NP-hard? En wanneer NP-volledig? (De definitie van $\mathcal{NP}$ hoeft niet te worden gegeven.)

Gegeven is nu dat (1) 3Kleur $\in \mathcal{NPC}$, (2) SAT $\in \mathcal{NPC}$ en (3) SAT $\leq_{\mathrm{P}}$ 3Kleur. Verder zagen we hierboven dat (4) CliqueCover $\in \mathcal{NP}$ en (5) 3Kleur $\leq_{\mathrm{P}}$ CliqueCover.

d. (12 punten)
Beantwoord de volgende vragen en geef bij elk antwoord een *duidelijke uitleg*.
(i) Toon aan dat er een polynomiale reductie bestaat van SAT naar CliqueCover. Welke van (1) t/m (5) heb je daarbij nodig?
(ii) Waarom geldt ook dat CliqueCover $\leq_{\mathrm{P}}$ 3Kleur?
(iii) Volgt uit (1) t/m (5) dat CliqueCover $\in \mathcal{NPC}$, of heb je daar de reductie uit (ii) voor nodig?
(iv) Wat weet je van 2Kleur?[3] Bestaat er een polynomiale reductie van 3Kleur naar 2Kleur?

[3] 2Kleur wordt analoog gedefinieerd als 3Kleur, maar nu moeten de knopen met hooguit twee i.p.v. met drie kleuren gekleurd worden.