Tentamen Complexiteit
Dinsdag 10 juli 2018, 14.00 – 17.00 uur

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

Opgave 1. (13 punten)
Gegeven een array A dat $n \geq 4$ verschillende getallen bevat (n is een viervoud). Verder geldt dat de eerste helft van A *oplopend* gesorteerd is, en hetzelfde geldt voor de tweede helft. De k-de waarde in grootte uit A is gedefinieerd als het arrayelement dat kleiner is dan precies $k-1$ andere arrayelementen, dus de k-de waarde van groot naar klein.
Gevraagd worden het $\frac{n}{4}$-de getal in grootte en het $\frac{3n}{4}+1$-de getal in grootte[1] uit A. Preciezer: we zoeken het paar (p,q) waarvoor geldt dat $A[p]$ de $\frac{n}{4}$-de waarde in grootte is en $A[q]$ de $\frac{3n}{4}+1$-ste waarde in grootte.
Voorbeeld met $n = 8$: $12, 20, 35, 42, 16, 22, 27, 30$. We zoeken de 2-de waarde in grootte en de 7-de waarde in grootte, dus $(p,q) = (3,5)$.

a. (3 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 kun je zeggen over de bladeren en wat stelt de hoogte van zo'n boom voor?
b. (6 punten)
Hoeveel verschillende antwoorden (dus paren (p,q) met p en q als boven) kunnen voorkomen bij invoerrijtjes van het bekeken type?
c. (4 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 $\lceil 2 \lg n \rceil - 2$ vergelijkingen moet doen in de worst case. Formuleer je bewijs zorgvuldig, maak gebruik van **a** en **b** en geef aan welke stellingen/eigenschappen je gebruikt.

Opgave 2. (9 punten)
Bekijk de klasse van graafalgoritmen die alleen vragen stellen van de vorm: "zit er een tak tussen knoop v en knoop w?". Bewijs met een adversary-argument dat elk algoritme uit die klasse dat checkt of een ongerichte graaf met n knopen ($n > 0$, even) samenhangend is, in de worst case altijd ten minste $\frac{n^2}{4}$ van die vragen moet stellen. Geef daartoe een adversary-strategie en leg uit waarom deze de gevraagde ondergrens oplevert.
Hint: voordat de adversary vragen van het algoritme beantwoordt splitst zij de knoopverzameling in twee disjuncte verzamelingen van elk $\frac{n}{2}$ knopen.

Opgave 3. (30 punten)
Gegeven is een array $A = A[1], A[2], \ldots, A[n]$, dat verschillende gehele getallen bevat, waarbij (*) op de oneven posities een waarde > 0 staat, en op de even posities een waarde < 0. Verder is $n = 2^k$ voor een geheel getal $k \geq 1$ (dus $n \geq 2$). We gaan het array in **a** sorteren met behulp van Shellsort, en in **b**/**c**/**d** met behulp van Mergesort.

[1] Dus het arrayelement dat kleiner is dan $\frac{3n}{4}$ andere arrayelementen, ofwel de $\frac{n}{4}$-de van klein naar groot.

a. (10 punten)
We bekijken Shellsort met sprongafstanden (= stapgroottes) $\frac{n}{2}, \frac{n}{2^2}, \frac{n}{2^3}, ..., 2, 1$.
Toon aan dat het aantal arrayvergelijkingen dat Shellsort met deze sprongafstanden doet op een array van het bovenbeschreven type $\geq \frac{n^2}{8}$ is.
Hint: let op de negatieve getallen in de laatste ronde.
b. (5 punten)
Leg uit hoe Mergesort werkt op het volgende voorbeeldarray: $5, -3, 3, -6, 7, -2, 8, -4$. Geef duidelijk de resultaten van de recursieve aanroepen aan. Hoeveel arrayvergelijkingen doet het algoritme op dit rijtje?
c. (5 punten)
Het aantal arrayvergelijkingen dat Mergesort doet op rijtjes van het gegeven type (*) in de *best case* noemen we $B(n)$. Leg uit waarom $B(n)$ voldoet aan de volgende recurrente betrekking. Leg vooral heel duidelijk uit hoe we aan de term $\frac{3}{4}n$ komen.

$$B(n) = \begin{cases} 1 & n = 2 \\ 2B(\frac{n}{2}) + \frac{3}{4}n & n \geq 4, n = 2^k \end{cases}$$

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

Opgave 4. (22 punten)
Gegeven is een array $A = A[1], A[2], \ldots, A[n]$ dat $n \geq 2$ verschillende gehele getallen bevat. Onderstaand algoritme bepaalt (de index van) het voorste array-element dat kleiner is dan alles wat erachter staat. In regel (2) en regel (4) geldt dat als de eerste test onwaar is de tweede test niet meer wordt gedaan.

```
(1)     i := 1; min := n; gevonden := False;
(2)     while (i < n) and not gevonden do
(3)         j := i + 1; doorgaan := True;
(4)         while (j ≤ n) and doorgaan do
(5)             if (A[i] < A[j]) then
(6)                 j := j + 1;
                else
(7)                 doorgaan := False;
                fi
            od
(8)         if j > n then
(9)             min := i; gevonden := True;
            fi
(10)        i := i + 1;
        od
(11)    return min;
```

Het vergelijken van array-elementen in regel (5) is maatgevend voor de complexiteit van het algoritme.

a. (3 punten)
Leg uit waarom dit algoritme (de index van) de kleinste waarde (minimum) uit A oplevert.

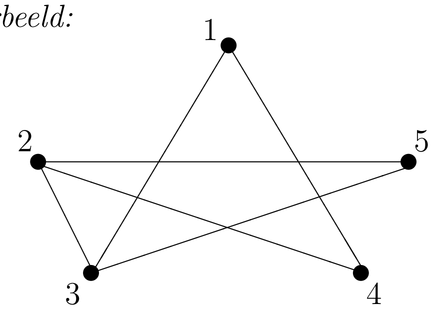
In **b** en **c** veronderstellen we dat het minimum zich bevindt op positie k met $1 \leq k \leq n$. Let op het (kleine) onderscheid tussen de gevallen $k = n$ en $k < n$ in het algoritme.

b. (8 punten)
(i) Hoeveel arrayvergelijkingen worden er minimaal gedaan? Druk het aantal arrayvergelijkingen uit in k en/of n (middels een gesloten formule).
(ii) Voor wat voor invoerrijtjes komt dat voor? Geef een karakterisering van *alle* gevallen. Leid daartoe uit het algoritme af wat moet gelden voor A opdat zo weinig mogelijk vergelijkingen gedaan worden.
(iii) Geef ten slotte twee concrete, illustratieve voorbeelden van best case invoerrijtjes met $n = 8$: het ene heeft het minimum op positie n, het andere op positie 5.
c. (8 punten)
(i) Hoeveel arrayvergelijkingen worden er maximaal gedaan? Druk het aantal arrayvergelijkingen uit in k en/of n (middels een gesloten formule).
(ii) Voor wat voor invoerrijtjes komt dat voor? Geef een karakterisering van *alle* gevallen. Leid daartoe uit het algoritme af wat moet gelden voor A opdat zo veel mogelijk vergelijkingen gedaan worden.
(iii) Geef weer twee concrete, illustratieve voorbeelden met $n = 8$, als in **b.**
d. (3 punten)
Voor welke k is het aantal arrayvergelijkingen uit **c.(i)** maximaal? Concludeer daaruit hoeveel arrayvergelijkingen het algoritme doet in de worst case en voor wat voor invoerrijtjes dat voorkomt.

Opgave 5. (26 punten)
We bekijken het volgende beslissingsprobleem:

VertexCover (VC): Gegeven een ongerichte graaf $\mathcal{G} = (V, E)$ en een positief geheel getal k.
Vraag: heeft $\mathcal{G}$ een *vertex cover* bestaande uit $\leq k$ knopen?
Een vertex cover is een deelverzameling V' van V zodat voor elke tak $(u, v) \in E$ minstens één van de uiteinden u of v bevat is in V'.

Voorbeeld:



Nevenstaande graaf $\mathcal{G}_1$ heeft een vertex cover ter grootte 3, namelijk $\{3, 4, 5\}$. $\langle \mathcal{G}_1, 3 \rangle$ is dus een ja-instantie van VC.

a. (10 punten)
Toon aan dat VC $\in \mathcal{NP}$ door een *niet-deterministisch polynomiaal* algoritme voor VC te geven. Het algoritme heeft dus als invoer een ongerichte graaf $\mathcal{G} = (V, E)$ en een geheel getal $k > 0$. Het algoritme moet "ja" opleveren dan en slechts dan als $x = \langle \mathcal{G}, k \rangle$ een ja-instantie is voor VC (&). Je mag er van uitgaan dat het aantal knopen van $\mathcal{G}$ bekend is en

dat $V = \{1, 2, \ldots, n\}$.
Leg uit wat in elke fase van het algoritme gebeurt *en hoe*, en hoeveel stappen dat kost. Leg ook uit waarom jouw algoritme aan (&) voldoet en waarom het polynomiaal is in $|x|$.

We definiëren het beslissingsprobleem VC3 als volgt:

VC3: Gegeven een ongerichte graaf $G = (V, E)$.
Vraag: heeft $\mathcal{G}$ een vertex cover ter grootte ≤ 3?

b. (5 punten)
Laat zien dat (1) VC3 $\in \mathcal{P}$ door in woorden een polynomiaal algoritme te geven. Geef ook aan waarom je algoritme polynomiaal 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.)

Van het werkcollege weten we dat (2) Kliek $\leq_{\mathrm{P}}$ VC. Ook is gemakkelijk in te zien dat (3) Kliek $\in \mathcal{NP}$. In **a** zagen we al dat (4) VC $\in \mathcal{NP}$.

d. (9 punten)
Beantwoord de volgende vragen en geef bij elk antwoord een *duidelijke uitleg*.
(i) Stel dat VC $\in \mathcal{NPC}$. Volgt nu uit (2) en (3) dat Kliek $\in \mathcal{NPC}$? Waarom wel/niet?
(ii) Stel dat Kliek $\in \mathcal{NPC}$. Volgt nu uit (2) en (4) dat VC $\in \mathcal{NPC}$? Waarom wel/niet?
(iii) Laat behalve (1) t/m (4) nu ook gegeven zijn dat Kliek $\in \mathcal{NPC}$ en veronderstel verder dat VC $\leq_{\mathrm{P}}$ VC3. Wat betekent dat voor Kliek? En wat betekent het voor $\mathcal{NP}$ in het algemeen?

EINDE