Tentamen Complexiteit
Donderdag 5 juni 2008, 14.00-17.00 uur

Geef een *duidelijke* toelichting bij al je antwoorden!

Opgave 1. (20 punten)
a. (5 punten)
Leg uit wat een beslissingsboom is voor algoritmen gebaseerd op arrayvergelijkingen: wat stellen de interne knopen en de bladeren voor, een pad van de wortel naar een blad, linker- en rechtersubboom van een knoop, de hoogte van de boom, etcetera.

Gegeven twee oplopend gesorteerde even lange rijen A en B met in totaal $2n$ getallen. Gevraagd wordt het n-de getal (in volgorde van klein naar groot) van de in totaal $2n$ elementen van A en B.
b. (5 punten)
Geef een algoritme dat dit probleem in maximaal n arrayvergelijkingen oplost door een merge toe te passen.
c. (5 punten)
Bewijs met behulp van een beslissingsboomargument dat *elk* algoritme dat dit probleem oplost m.b.v. arrayvergelijkingen ten minste $\lceil \lg n \rceil + 1$ vergelijkingen moet doen in de worst case.
d. (5 punten)
We zoeken nu de n-de waarde in grootte zoals boven, maar de ene gesorteerde rij bevat $\frac{n}{2}$ elementen en de andere $\frac{3n}{2}$. Net als in **c.** kunnen we een ondergrens voor de worst case bewijzen voor het probleem met dit soort invoerrijtjes. Wat verandert er dan in het bewijs van **c.** en welke ondergrens levert dat op?

Opgave 2. (15 punten)
a. (10 punten)
Bewijs dat het aantal vergelijkingen dat *Shellsort* met sprongafstanden (= stapgroottes) $n/3$, $n/9$, $n/27$, ..., 9, 3, 1, in de worst case doet $\Omega(n^2)$ is. Neem aan dat n een macht van 3 is.
b. (2 punten)
Wat betekent het als een sorteermethode stabiel is?
c. (3 punten)
Is Shellsort met sprongafstanden bijv. zoals hierboven stabiel? Licht je antwoord toe.

Opgave 3. (15 punten)
Gegeven is een array A met $n = 2^k$ ($k \geq 1$) verschillende gehele getallen. Op de oneven posities staan negatieve getallen, op de even posities positieve.
De bedoeling is om alle negatieve elementen voor alle positieve te zetten, waarbij de onderlinge volgorde van de negatieve, respectievelijk positieve, elementen blijft gehandhaafd. We doen dit als volgt. Voor $n > 2$ reorganiseren we *recursief* de eerste en tweede helft van A zoals hierboven bedoeld. Vervolgens moeten we nog enige elementen verwisselen om het hele array in de juiste vorm te krijgen. Laat $V(n)$ het aantal verwisselingen zijn dat dit algoritme doet.

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

$$V(n) = \begin{cases} 0 & n = 2 \\ 2V(\frac{n}{2}) + \frac{n}{4} & n \geq 4,\ n = 2^k \end{cases}$$

Leg ook uit hoe je aan de beginvoorwaarde en de term $\frac{n}{4}$ komt.
b. (10 punten)
Los de recurrente betrekking uit **a.** op door deze herhaald in zichzelf in te vullen en bewijs met behulp van volledige inductie dat de aldus gevonden oplossing (uitgedrukt in n) inderdaad voldoet.

Opgave 4. (25 punten)
Gegeven is een array A met $n = 2^k$ ($k \geq 1$, geheel) verschillende gehele getallen $A[1]$, $A[2]$, ..., $A[n]$. We bekijken een algoritme dat het kleinste *en* het grootste getal uit het array oplevert.

```
(1)   d := 1;
(2)   while d < n do
(3)       i := 1;
(4)       while i ≤ n do
(5)           if A[i] > A[i + d] then
(6)               wissel(A[i], A[i + d]);
(7)           fi
(8)           if d ≠ 1 then
(9)               if A[i + d − 1] > A[i + 2d − 1] then
(10)                  wissel(A[i + d − 1], A[i + 2d − 1]);
(11)              fi
(12)          fi
(13)          i := i + 2 ∗ d;
(14)      od
(15)      d := 2 ∗ d;
(16)  od
```

De functie *wissel* verwisselt de waarden van zijn argumenten. Na afloop zitten in $A[1]$ en $A[n]$ het kleinste respectievelijk het grootste van de oorspronkelijke array-elementen.

a. (4 punten)
Neem aan dat het algoritme (voor $n = 16$) reeds gevorderd is tot een situatie waarin $d = 2$, en de inhoud van het array A op dat moment is:

$$9, 14, 8, 12, 4, 5, 16, 22, 11, 21, 7, 13, 3, 18, 10, 19.$$

Het algoritme gaat nu aan de slag met regel (3) en verder. Voer het algoritme uit en geef enkele relevante tussenstappen, bijv. de inhoud van het array A steeds wanneer regel (13) wordt uitgevoerd.
b. (6 punten)
Laat zien dat het aantal arrayvergelijkingen (regel (5) en (9) samen) maatgevend is voor de complexiteit van het algoritme. Doe dit door het aantal keren dat een regel wordt

gedaan in verband te brengen met het aantal keren dat de test in regel (5) of de test in regel (9) wordt uitgevoerd.
c. (5 punten)
Hoeveel arrayvergelijkingen (regel (5) en (9) samen) worden precies gedaan?
Bepaal daartoe eerst hoeveel vergelijkingen per doorgang door de d-loop (buitenste while), worden gedaan, dus voor respectievelijk $d = 1, 2, 4, \ldots, \frac{n}{2}$. Je mag verder gebruiken dat $\sum_{i=1}^{k-1}(\frac{1}{2})^i = 1 - (\frac{1}{2})^{k-1}$. Druk je antwoord uit in n.

We bekijken nu het aantal verwisselingen (regel (6) en (10) samen) dat het algoritme doet. Dit aantal is in de best case 0 en in de worst case gelijk aan het aantal arrayvergelijkingen.
d. (5 punten)
Bekijk de gevallen $n = 2$ en $n = 4$. Beschrijf voor wat voor rijtjes het algoritme het minimale aantal verwisselingen doet en voor wat voor rijtjes het maximale aantal. Geef een duidelijke uitleg waarom dit alle best/worst case invoerrijtjes zijn. Geef tevens een paar concrete voorbeelden.
e. (5 punten)
We bekijken nu alleen de best case, en wel voor $n = 8$. Geef een beschrijving (met uitleg) van de best case invoerrijtjes (ook weer alle gevallen). Geef tevens drie verschillende illustratieve voorbeelden van zulke rijtjes.

Opgave 5. (25 punten)
We hebben de volgende twee beslissingsproblemen:

3SAT: Gegeven een logische formule ϕ in 3-CNF (dus een AND van clausules waarbij elke clausule de OR van *precies drie* verschillende literals is).
Vraag: is er een waardering (waarheidstoekenning) van de in ϕ voorkomende variabelen x_i, zodanig dat ϕ wordt waargemaakt, dat wil zeggen dat per clausule minstens één literal (dat is een x_i of een $\neg x_i$) waar is?
Opmerking: zo'n waardering zullen we even een T-waardering noemen.

TFSAT: Gegeven is een logische formule ϕ in 3-CNF.
Vraag: is er een waardering die per clausule minstens één literal true maakt en minstens één literal false?
Opmerking: zo'n waardering zullen we hier een TF-waardering noemen.

a. (10 punten)
Toon aan dat TFSAT $\in \mathcal{NP}$ door een *niet-deterministisch polynomiaal* algoritme voor TFSAT te geven. Het algoritme heeft dus als invoer een logische formule ϕ in 3-CNF en moet "ja" opleveren dan en slechts dan als ϕ een ja-instantie is (&). Leg onder andere uit wat in elke fase van het algoritme gebeurt, en in het bijzonder wat in fase 2 wordt gecontroleerd en hoe, en hoeveel stappen dat kost. Leg ook uit waarom je algoritme aan (&) voldoet en waarom het polynomiaal is.

We bekijken een transformatie T van instanties van 3SAT naar instanties van TFSAT.
Gegeven is ϕ, een logische expressie in 3-CNF. Noem de in ϕ voorkomende logische variabelen $x_1, x_2, \ldots, x_n$. We construeren hierbij een logische formule $T(\phi)$, eveneens in 3-CNF. Voor elk van de clausules van ϕ maken we twee nieuwe clausules. De conjunctie van deze clausules is dan $T(\phi)$. We doen dit als volgt: de j-de clausule $(l_j^1 \vee l_j^2 \vee l_j^3)$ wordt overgezet

in $(l_j^1 \vee l_j^2 \vee y_j) \wedge (\neg y_j \vee l_j^3 \vee z)$. Alle variabelen y_j (één per clausule van ϕ) en de variabele z (hetzelfde voor elke clausule van ϕ) zijn nieuw gekozen.
Het is duidelijk dat de transformatie van $T(\phi)$ uit ϕ polynomiaal is **(*)**.

Voorbeeld: De formule $\phi = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_3 \vee x_2)$ wordt afgebeeld op $T(\phi) = (x_1 \vee x_2 \vee y_1) \wedge (\neg y_1 \vee x_3 \vee z) \wedge (\neg x_1 \vee \neg x_3 \vee y_2) \wedge (\neg y_2 \vee x_2 \vee z)$.

In **b.** en **c.** wordt aangetoond dat geldt: ϕ is een ja-instantie van 3SAT $\Longleftrightarrow$ $T(\phi)$ is een ja-instantie van TFSAT. Ofwel: er bestaat een T-waardering van de $x_1, \ldots, x_n$ $\Longleftrightarrow$ er bestaat een TF-waardering van de in $T(\phi)$ voorkomende variabelen.
Samen met **(*)** volgt hier dan uit: 3SAT $\leq_P$ TFSAT **(**)**

b. (4 punten)
Laat zien: als er een T-waardering bestaat van $x_1, \ldots, x_n$ dan bestaat er een TF-waardering van de in $T(\phi)$ voorkomende variabelen.
c. (4 punten)
Laat zien: als er een TF-waardering van de in $T(\phi)$ voorkomende variabelen bestaat dan bestaat er een T-waardering van $x_1, \ldots, x_n$. Merk op: als w een TF-waardering is voor $T(\phi)$, dan is de tegengestelde waardering $\overline{w}$ (gedefinieerd als: $\overline{w}(x) = \text{true} \Leftrightarrow w(x) = \text{false}$, x een logische variabele) dat ook. Gebruik dit om het gevraagde aan te tonen.
d. (2 punten)
Wanneer is een beslissingsprobleem P NP-hard? En wanneer NP-volledig? (Geef de definitie van NP-hard/ NP-volledigheid; de definitie van NP hoeft niet te worden gegeven.)
e. (5 punten)
Welk van de twee onderstaande beweringen is juist? Motiveer je antwoord en formuleer duidelijk eventueel gebruikte stellingen.
(i) Stel dat TFSAT $\in \mathcal{NPC}$. Volgt dan uit **a.** en **(**)** dat 3SAT ook NP-volledig is ?
(ii) Stel dat 3SAT $\in \mathcal{NPC}$. Volgt dan uit **a.** en **(**)** dat TFSAT ook NP-volledig is?

Veel succes!