Proeftentamen IN3105
Complexiteitstheorie
Uitwerkingen

Notationele conventies:

- $A \leq B$ betekent dat er een polynomiale-tijd reductie van A naar B bestaat.
- A^c is het complement van het probleem A.
- Er wordt steeds vanuit gegaan, dat **P** ≠ **NP** en **NP** ≠ **co-NP**, tenzij uitdrukkelijk het tegendeel wordt vermeld.

MEERKEUZEVRAGEN

Vraag 1. Als A een probleem in **NPC** is, dan geldt dat

a. A geen polynomiaal algoritme heeft, tenzij **P = NP.**
b. yes-instanties en/of no-instanties van A polynomiaal verifieerbaar zijn.
c. als A ook een co-**NP**-probleem is, geldt dat **NP** = **co-NP**.
d. voor elk **co-NP** probleem B geldt $B \leq A^c$.

Antwoord

Alternatief a. is juist, want we veronderstellen P≠ NP. Alternatief b is onjuist: alleen yes-instanties van A zijn polynomiaal verifieerbaar . Alternatief c. is juist: als NPC en co-NP een niet-lege doorsnede hebben, geldt NP = co-NP.
Alternatief d. is juist: als A een NPC probleem is, is A^c een co-NPC probleem.

Vraag 2. Er zijn 3 beslissingsproblemen A, B en C. Het probleem A is een **co-NP** probleem, B is een **NP**-probleem en C is een **NP**-hard probleem. Er geldt nu

a. Als $A \leq C$, dan geldt **NP = co-NP**.
b. Als $C \leq A$, dan geldt **P = NP.**
c. $A \leq B$ of $B \leq A$.
d. $B \leq C$.

Antwoord

Alternatief a. is onjuist, want A kan bv. een **P**-probleem zijn en dus een **NP**-probleem
Alternatief b. is ook onjuist: alleen **NP = co-NP** volgt als $C \leq A$.
Ook alternatief c. is onwaar: neem maar eens voor A een probleem zonder yes-instanties en B een probleem zonder no-instanties. Alternatief d. is juist op grond van de definitie van NP-harde problemen.

Vraag 3. Welke van de volgende uitspraken is/zijn waar?

a. alle **P**-problemen zijn polynomiaal reduceerbaar tot elkaar.
b. alle **PSPACE**-problemen zijn polynomiaal reduceerbaar tot elkaar.
c. alle **NPC**-problemen zijn polynomiaal reduceerbaar tot elkaar.
d. alle **NP**-harde problemen zijn polynomiaal reduceerbaar tot elkaar.

Antwoord a. is onjuist: neem maar eens een probleem met een lege verzameling yes-instanties en een probleem met een lege verzameling no-instanties; om dezelfde reden is b. onjuist, aangezien PSPACE de klasse P omvat.
Alternatief c. is per definitie van NPC juist. Alternatief d. is onjuist: NP-harde problemen behoeven niet polynomiaal gerelateerd te zijn.

Vraag 4. Sipser definieert beslissingsproblemen als talen over een alfabet. Een beslissingsprobleem is in zijn notatie een verzameling A van yes-instanties van het probleem. Als A een **NP**-probleem is en B een **P**-probleem dan is volgens de definitie van Sipser:

a. $A^c \cap B$ een **NP**-probleem.
b. $A \cup B^c$ een **NP**-probleem.
c. $A \times B$ een **NP**-probleem.
d. $A^c \cup B^c$ een **co-NP**-probleem.

Antwoord Bedenk, dat A^c een co-NP probleem is en B^c een P-probleem. Alternatief a. formuleert een P-probleem. Dit is natuurlijk ook een NP-probleem: a. is juist. Alternatief b. geeft een NP-probleem: b. is juist. Omdat NP ook gesloten is onder cartesisch product geldt c. is juist. Alternatief d. is natuurlijk ook juist, want co-NP (evenals NP) is gesloten onder vereniging (ga na).

Vraag 5. Het 3-falsificatie-probleem bestaat eruit om, gegeven een verzameling 3-SAT clauses C over een verzameling U van propositie atomen, te bepalen of er een waarheidstoekenning τ bestaat die *alle* clauses in C *onwaar* maakt.
Professor W. Tutbeter zegt een algoritme te hebben ontworpen om dit probleem op te lossen:

```
input: U = {U1,...,Un}, C = {C1, C2, ... , Cm}
begin
   List := ∅ ; i := 0 ; Falsifiable := true;
   while i ≠ m and Falsifiable
        i := i+1;
        let Ci = {x1,x2,x3};
        for j=1 to 3 do
           if negated(xj)∈ List
              then Falsifiable:= false
           else List := List ∪ { xj };
   return Falsifiable;
end
```

Hierin is `negated(`x_j`)` een procedure die de ontkenning van de literal x_j produceert, d.w.z. als $x_j \in U$ dan geldt `negated(`x_j`)` $= \neg x_j$ anders is $x_j = \neg u_j$ voor een $u_j \in U$ en geldt `negated(`x_j`)` $= u_j$.
Uit dit algoritme blijkt dat

a. het bovengenoemde probleem een **P**-probleem is.
b. het bovengenoemde algoritme alleen nagaat of er clauses voorkomen waarin een propositieatoom u en zijn genegeerde $\neg u$ voorkomen; dit zegt echter niets over het onwaar kunnen maken van *alle* clauses.
c. het algoritme moet wel incorrect zijn, omdat het falsificatieprobleem niet in polynomiale tijd is op te lossen tenzij **P** = **NP**.
d. het falsificatieprobleem is het complement van het satisfiability probleem; daarom is het een **co-NP** compleet probleem; bovenstaand algoritme kan daarom niet correct zijn.

Antwoord Bovengenoemd algoritme gaat na of er een u in U voorkomt waarvoor geldt dat zowel u als $\neg u$ voorkomt in de verzameling van clauses C. Als dit gebeurt moet iedere waarheidstoekenning minstens één clause waarmaken in C (de clause waarin u voorkomt of de clause waarin $\neg u$ voorkomt) en is C niet falsificeerbaar. Als dit niet gebeurt (Falsifiable = true) dan is het mogelijk alle literals onwaar te maken en zijn alle clauses daarmee onwaar te maken. De looptijd van het algoritme is kwadratisch in het aantal clause literals en dus een polynomiaal algoritme: alleen alternatief a. is waar.

Vraag 6. Iemand beweert het volgende:

Het is bekend dat VERTEXCOVER (VC) een 2-approximatie-algoritme heeft; omdat VC een **NPC** probleem is, volgt hieruit dat ieder **NP**-probleem A polynomiaal reduceerbaar is naar VC. Maar dan geldt onmiddelllijk, dat ieder **NP**-probleem een 2-approximatie-algoritme heeft.

Er geldt nu:

a. Deze bewering is onwaar, omdat VC geen 2-approximatie algoritme heeft.
b. Deze bewering is onwaar, omdat VC geen NPC probleem is.
c. Deze bewering is onwaar, omdat polynomiale tijdreducties niet approximatie behoudend zijn.
d. Deze bewering is waar.

Antwoord de optimaliseringsvariant van VC heeft wel degelijk een 2-approximatie algoritme; alternatief a. is onjuist. VC is een NPC probleem, alternatief b. is ook onjuist. Alternatief c. is onjuist omdat inderdaad polynomiale tijdreducties (bv van VC naar TSP) niet approximatie-behoudend zijn. Daarom is alternatief d. niet waar.

Vraag 7. A, B en C zijn beslissingsproblemen.
Als A $\leq$ B en B is een restrictie van C dan geldt:

a. Als A sterk **NP**-volledig is, dan is C ook sterk **NP**-volledig.
b. Als C een pseudo-polynomiaal algoritme heeft, dan heeft B ook een pseudo-polynomiaal algoritme.
c. Als C een sterk-**NP**-volledig probleem is, dan is B een **NP**-volledig probleem.
d. Als B een **NP**-volledig probleem is, dan is C ook een **NP**-volledig probleem.

Antwoord Alternatief a. is onjuist: C behoeft geen NP-probleem te zijn. Alternatief b. is juist: als de eigenschap geldt voor alle instanties van C dan geldt zij zeker voor de subset van instanties die het probleem B vormen. Alternatief c. is onjuist. Het is heel goed mogelijk dat B als restrictie van C een polynomiaal probleem is en C een sterk NP-volledig probleem. Alternatief d. is onjuist: C is minstens zo moeilijk als B en is een NP-hard, maar niet noodzakelijk een NP-probleem.

Vraag 8. Het ALARMCENTRALE probleem bestaat uit een graaf $G = (V, E)$, een afstand $d \in Z^+$ en een budget $B \in Z^+$. De vraag is of er een subset V' van V (locaties van centrales) bestaat met $|V'| \leq B$, zodat voor geen enkele knoop v uit V de afstand tot alle centrales in V' groter is dan d, d.w.z. voor iedere knoop $v \in V$ is er een element $v' \in V'$ te vinden zodanig dat er een pad van v naar v' in G bestaat ter lengte van hoogstens d.
Voor dit probleem geldt het volgende:

a. instanties van dit probleem met $d \leq 1$ zijn in polynomiale tijd oplosbaar;
b. instanties van dit probleem met $d = |V|$ zijn in polynomiale tijd oplosbaar;
c. instanties van dit probleem met $d=1$ zijn vertex cover instanties;
d. instanties van dit probleem met $B \leq 2$ zijn in polynomiale tijd oplosbaar.

Antwoord *Alternatief a.* is niet juist: dit zijn precies alle vertex-cover instanties; *alternatief c.* is dus juist. *Alternatief b.* is juist: je kunt voor alle paren van knopen (v,w) de lengte van het kortste pad tussen v en w bepalen. Deze lengte is ofwel ∞ of $< |V|$. Beschouw nu een willekeurige opsomming $(v_1, v_2, \ldots, v_n)$ van de knopen in V en beschouw ze alle als vrij. Zolang B > 0 is, wijs je een centrale aan de eerste vrije knoop toe, schrap je alle locaties die op afstand <|V| van deze locatie liggen, en verlaag je B met 1. Als alle locaties geschrapt zijn heb je een ja-instantie, anders een nee-instantie.
Alternatief d. is juist: er zijn in totaal $O(|V|^3)$ verzamelingen van drie locaties te checken; zo'n check kan in polynomiale tijd plaatsvinden.

Vraag 9. Men wil nagaan of in een array A van n getallen er een getal is dat meer dan $n/2$ maal voorkomt. Het volgende randomized algoritme wordt hiervoor gebruikt.

```
majority(A, n)                    % A is een array met n elementen
   begin
        index:= random(1,n);  % kies een random getal uit 1..n
        x := a[index];
        count := 0;
        for k=1 to n do
              if a[k] = x then count := count+1
              k:=k+1;
        return (count > n/2);
    end
```

Er geldt nu dat:

a. er een randomized algoritme bestaat dat in $O(n)$-tijd altijd een correct antwoord geeft als A een waarde bevat die meer dan $n/2$-maal voorkomt.
b. voor elke positieve integer k er een randomized algoritme bestaat met polynomiale looptijd dat met kans $< 0.5^k$ een incorrect antwoord op het bovenstaande probleem geeft.
c. dit probleem tot de klasse **RP** behoort.
d. dit probleem tot de klasse **BPP** behoort.

Antwoord Dit randomized algoritme kan alleen een foutieve uitslag geven als er een getal in A is dat vaker dan n/2 maal voorkomt, terwijl het algoritme een ander getal selecteert en als uitslag "false" (NO) oplevert. Deze kans is echter kleiner dan 0.5. Alternatief a. is derhalve onjuist. Als het algoritme

k-maal herhaalt wordt is de kans dat ten onrechte k-maal “false” wordt opgeleverd kleiner dan 0.5^k: alternatief b. is juist. Alternatief c. en alternatief d. zijn ook beide juist omdat het probleem een RP probleem is (kans op ten onrechte NO antwoord is hoogstens 0.5 en kans op ten onrechte “YES” antwoord is 0) en en RP een subset is van BPP.

Vraag 10. L is een taal en M is een probabilistische polynomiale Turingmachine voor L. Welke van de volgende beweringen laat toe de conclusie te trekken dat $L \in$ **BPP**?

a. Pr [M accepteert x | $x \in L$] $\geq .45$ en Pr [M accepteert x | $x \notin L$] $\leq .55$
b. Pr [M accepteert x | $x \in L$] $\geq .55$ en Pr [M accepteert x | $x \notin L$] $\leq .45$
c. Pr [M accepteert x | $x \in L$] $\geq .95$ en Pr [M accepteert x | $x \notin L$] $\geq .95$
d. Pr [M accepteert x | $x \in L$] $\geq .9$ en Pr [M accepteert x | $x \notin L$] $\geq .09$

Antwoord Pas definitie toe. Alternatief a. is onwaar; alternatief b is waar; alternatief c is onwaar; alternatief d is ook onwaar.

OPEN VRAGEN I

Onderstaande vragen dient u kort te beantwoorden. Een antwoord dient in hoogstens 5 regels gegeven te worden.

Vraag 11. Geef precies aan, waarom nooit kan gelden dat **NPC = P**, zelfs al zou gelden dat **P = NP**.

Antwoord Omdat de lege taal $\varnothing$ en de volledige taal Σ^* nooit als beeld van een polynomiale tijdreductie kunnen optreden voor een probleem in P dat zowel "yes" als "no"-instanties heeft, zijn deze problemen nooit NP-volledig, ook niet als P=NP.

Vraag 12. Stel, dat SAT oplosbaar blijkt te zijn in $O(n^2 \log n)$. Geef een overtuigend argument voor of tegen de bewering dat er dan een constante k bestaat zodat ieder NP-probleem oplosbaar is in $O(n^k)$.

Antwoord Als ieder probleem in dit geval oplosbaar zou zijn in $O(n^k)$ dan zouden er volgens de tijdcomplexiteitshierarchie P-problemen zijn die in $O(n^{k+1})$ –tijd kunnen worden opgelost, maar niet in $O(n^k)$. Dit is in tegenspraak met de conclusie dat alle problemen in NP, en dus ook alle problemen in P, kunnen worden opgelost in $O(n^k)$.

Vraag 13. Het MIN-BINPACKING-probleem is het volgende probleem:
Instantie: Een n-tal positieve integers $a_1, a_2, \ldots, a_n$ en een positieve integer C (de capaciteit per bin). We mogen er hierbij vanuit gaan dat $a_j \leq C$ voor iedere $j = 1, \ldots N$.
Vraag: Wat is het minimaal benodigd aantal bins om de n items op te bergen zonder de capaciteit van de bins te overschrijden?

Waarom kan er geen voor dit probleem geen approximatie-algoritme bestaan, dat dit probleem oplost met approximatieratio < 1.5, tenzij P=NP?

Antwoord Als zo'n approximatiealgoritme A bestaat, dan kan PARTITION in polynomiale tijd opgelost worden. Stel x is een yes-instantie van PARTITION, dan maken we een BINPACKING instantie met $C = \lfloor \sum a_i \rfloor/2$. Omdat A minder dan 1.5 keer het optimale aantal bins (=2) zal geven, retourneert A een getal $c < 3$. Als x een no instantie is, is het optimale aantal bins ≥ 3 en zal A een uitkomst $c \geq 3$ geven. Hiermee wordt in polynomiale tijd een NPC probleem opgelost.

Vraag 14. Geef zo nauwkeurig mogelijk de relaties (=, $\subset$ of $\subseteq$) tussen de volgende klassen: **P, NP, EXP, PSPACE, NPSPACE**.

Antwoord P $\subseteq$ NP $\subseteq$ PSPACE = NPSPACE $\subseteq$ EXP. Bovendien geldt: P $\subset$ EXP.

Vraag 15. Welke argumenten kun je aanvoeren voor de conclusie dat quantum algoritmen voor backtracking niet gebruikt kunnen worden om NP-complete problemen efficiënt op te lossen?

Antwoord Het algoritme van Grover is een methode om via brute force zoekmethoden zoals backtracking NPC problemen op te lossen. Vazirani cs hebben aangetoond dat hoogstens een kwadratische versnelling met kwantum methoden kan worden bereikt.

OPEN VRAGEN

Vraag 16. (10 punten)

Een probleem behoort tot de klasse NP^{PSPACE} als er een niet-deterministisch polynomiaal algoritme bestaat dat het probleem kan oplossen door gebruik te maken van een PSPACE-orakel.

Laat zien dat $NP^{PSPACE} = P^{PSPACE} = PSPACE$.

Antwoord:

$PSPACE \subseteq P^{PSPACE}$: ieder PSPACE-algoritme kan gezien worden als een polynomiaal algoritme dat gebruik maakt van een PSPACE-subroutine. Omdat $P \subseteq NP$ geldt tevens $P^{PSPACE} \subseteq NP^{PSPACE}$.
Hieruit volgt: $PSPACE \subseteq P^{PSPACE} \subseteq NP^{PSPACE}$ (1)
Neem nu een $PSPACE^{PSPACE}$ probleem. Dit kan worden opgelost in PSPACE: Een PSPACE-orakel aanroep kan worden vervangen door een subroutine die polynomiale ruimte vergt. Deze ruimte kan onmiddellijk gereclaimed worden. Derhalve geldt $PSPACE^{PSPACE} \subseteq PSPACE$. Omdat $NP \subseteq PSPACE$ geldt tevens $NP^{PSPACE} \subseteq PSPACE^{PSPACE}$ en daarom volgt:
$NP^{PSPACE} \subseteq PSPACE$. (2)
De gevraagde gelijkheden volgen nu onmiddellijk uit (1) en (2)

Vraag 17. (10 punten)

Gegeven zijn $N > 1$ DVD's en $M \geq 1$ mp3-files. Van iedere DVD D is de opslagcapaciteit in bytes bekend en van iedere mp3-file f de omvang in bytes. Gevraagd wordt of de N DVD's voldoende opslagcapaciteit hebben om de M files op te slaan. Het is hierbij niet toegestaan een mp3-file f op te splitsen en de gedeelten op meer dan één DVD op te slaan.

a. Kies een geschikt **NPC**-probleem A en formuleer de reductie van dit probleem A naar bovengenoemd probleem (≤ 6 regels).
U mag hierbij kiezen uit de volgende **NP**-complete problemen:
SAT, VC, PARTITION, CLIQUE, SUB(SET)SUM, HAMPAD.

b. Bewijs de correctheid van deze reductie (≤ 15 regels).

Antwoord:

a. Neem de volgende reductie f van PARTITION naar het opslagprobleem: $f(A,g) = (CD, F, o, b)$, waarbij $CD = \{cd1, cd2\}$, $F = A$, $o(cd1) = o(cd2) = \lfloor \Sigma_A\, g(a_i) / 2 \rfloor$ en $b(f_i) = g(a_i)$. Hierin is CD de verzameling CD's, F de verzameling mp3-files, o: CD → N de opslagcapaciteit en b: F → N de file grootte.

b. (1) Stel I is een yes-instantie. Dan bestaat er een subset $A1 \subseteq A$ zodat $g(A_1) = g(A\text{-}A_1) = \Sigma_A\, g(a_i) / 2 = \lfloor \Sigma_A\, g(a_i) / 2 \rfloor$. Maar dan geldt ook dat $f(I)$ een yes-instantie is: Kies $F_1 = A_1$ en $F_2 = A\text{-}A_1$. Dan kan F_1 op cd1 gebrand worden en de rest op cd2. Immers, $b(F_1) = g(A_1) = \lfloor \Sigma_A\, g(a_i) / 2 \rfloor = b(F_2)$. (2) Stel $f(I)$ is een yes-instantie, dan bestaat er een subset F_1 zodat $b(F_1) = b(F\text{-}F_1) = \lfloor \Sigma_F\, b(f_i) / 2 \rfloor$ en $b(F_1) + b(F\text{-}F_1) = \Sigma_F\, b(f_i)$. Hieruit volgt: $\lfloor \Sigma_F\, b(f_i) / 2 \rfloor = \Sigma_F\, b(f_i) / 2$. Maar dan is de oorspronkelijke instantie I ook een yes-instantie: kies maar $A_I = F_1$. Er geldt dan $g(A_1) = \Sigma_A\, g(a_i) / 2 = g(A\text{-}A_1)$. (3) De reductie is polynomiaal: CD wordt bepaald in O(1)-tijd, F in O(|A|) tijd, o en b worden berekend in O(|g|)-tijd. Totaal kost de berekening van f dus O(|I|)-tijd.