Uitwerkingen Tentamen TI3300

Complexiteitstheorie

15 april 2014

9.00 - 12.00 uur

- Dit tentamen bestaat uit 10 meerkeuzevragen, 5 korte (open) vragen en 2 open vragen.
- Voor de meerkeuzevragen kunt u maximaal 50 punten behalen, voor de korte open vragen maximaal 30 punten en voor de open vragen maximaal 20 punten.
- Het totaal aantal punten voor de meerkeuzevragen (*mp*) wordt als volgt bepaald:
  - Per meerkeuzevraag *i* wordt het aantal foutief beantwoorde alternatieven f_i bepaald. Aangezien per vraag 0,1,2,3 of 4 alternatieven juist kunnen zijn geldt: $0 \leq f_i \leq 4$;
  - Laat $f = \sum f_i$ voor alle 10 meerkeuzevragen ($0 \leq f \leq 40$) ; het aantal punten voor de meerkeuzevragen (*mp*) is dan $mp = 50 \times \max\{ 0, (40 - 1.5 \times f) / 40 \}$.
- Het eindcijfer *c* wordt bepaald door de som te nemen van *mp*, het totaal aantal punten behaald voor de 5 korte vragen (*kv*) en de som van het aantal punten behaald voor de twee open vragen (*ov*) en vervolgens het resultaat te delen door 10: $c = (mv+kv+ov)/10$; dit cijfer wordt vervolgens afgerond op het dichtstbijzijnde halve of gehele getal.

- Het gebruik van dictaat, aantekeningen, boek of andere bronnen is niet toegestaan.
- Het gebruik van grafische en niet-grafische rekenmachines is eveneens niet toegestaan.

- Beantwoord de meerkeuzevragen op het bijgeleverde antwoordformulier.
- Beantwoord de open vragen zo bondig mogelijk; geef geen irrelevante informatie, dit kan leiden tot puntenaftrek!
- Controleer of u op ieder blaadje uw naam en studienummer heeft vermeld en geef het totaal aantal ingeleverde bladen op (tenminste) de eerste pagina.

Notationele conventies:

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

MEERKEUZEVRAGEN (50 PUNTEN)

Vraag 1. Welke van de volgende uitspraken is waar?

a. SAT $\leq$ PATH.
b. PARTITION $\leq$ VERTEX COVER.
c. A $\leq$ TQBF voor elk **NP**-probleem A.
d. A $\leq$ B als A een willekeurig **P**-probleem is en B een **NP-hard** probleem.

antwoord a. is onjuist: PATH is in **P**, SAT is een **NPC**-probleem en onder de veronderstelling dat **P** $\neq$ **NP** kan er geen polynomiale reductie van SAT naar PATH bestaan.
b. is juist, beide zijn **NPC**-problemen.
c. is juist, immers **NP** $\subseteq$ **PSPACE** en elk **PSPACE**-probleem is polynomiaal reduceerbaar naar TQBF.
d. is juist, omdat **P** $\subseteq$ **NP** en ieder **NP**-probleem is polynomiaal reduceerbaar naar een **NP-hard** probleem

Vraag 2. Gegeven zijn de beslissingsproblemen A, B en C met C $\leq$ A en B $\leq$ C.

Het is bekend dat B een **NP-hard** probleem is.
Onder welke condities is het zeker dat B een **NPC**-probleem is?

a. als C een **NP**-probleem is.
b. als A een **NPC**-probleem is.
c. als A $\leq$ B.
d. als A een **NP**-probleem is.

antwoord a. is juist omdat B $\leq$ C en **NP** gesloten is onder $\leq$.
b. is juist omdat **NPC** $\subseteq$ **NP** en **NP** gesloten is onder $\leq$.
c. niet juist: A, B en C zijn dan even moeilijk, maar behoeven geen **NP**-problemen te zijn.
d. is juist: B $\leq$ C $\leq$ A impliceert B $\leq$ A en **NP** is gesloten onder $\leq$.

Vraag 3. Als **NP** $\neq$ **coNP**, dan geldt**:**

a. **P** $\neq$ **NP**.
b. **NPC** $\neq$ **coNPC**.
c. **NP** - **P** $\neq \varnothing$.
d. **coNP** $\cap$ **NP** $\neq \varnothing$.

antwoord a. is juist: omdat **P** = **coP**, impliceert **P** = **NP** dat **NP** = **coNP**.
b. is juist: zelfs **NPC** $\cap$ **coNPC** $\neq \varnothing$ zou **NP** = **coNP** impliceren.
c. is juist, als **NP** - **P** = $\varnothing$ zou **P** = **NP** en daarmee **NP** = **coNP**.
d. is juist, in alle gevallen geldt immers $\varnothing \neq$ **P** $\subseteq$ **coNP** $\cap$ **NP.**

Vraag 4. A, B en C zijn drie beslissingsproblemen, waarbij gegeven is dat B een **NP**-probleem is.
Als A $\leq$ B en B $\leq$ C, dan geldt:

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

antwoord a. is onjuist: we kunnen afleiden dat C een **NPC**-probleem is, maar $\leq$ bewaart geen sterke **NP**-volledigheid, zie bv. CLIQUE $\leq$ PARTITION.
b. is onjuist, B en C kunnen sterk-NP-volledige problemen zijn, zie bv. PARTITION $\leq$ CLIQUE
c. is onjuist, we kunnen niet hieruit afleiden dat B $\in$ **NPC (**Neem bv. maar eens B $\in$ **P**).
d. is juist, omdat **NP** gesloten is onder $\leq$.

Vraag 5. Welke van de onderstaande inclusierelaties zijn geldig?

a. **PSPACE** $\subset$ **NPC.**
b. $\mathbf{P}^{\mathbf{NP}} \subseteq \mathbf{P}^{\mathbf{coNP}}$.
c. **NPC** $\subseteq$ **PP**.
d. **P** $\subseteq$ **BPP**.

antwoord a. is niet juist; **PSPACE** omvat bv. **P** en **P** kan niet een strikte subset van **NPC** zijn.
b. is juist, er geldt zelfs $\mathbf{P}^{\mathbf{NP}} = \mathbf{P}^{\mathbf{coNP}}$**.**
c. is juist, want **NP** $\subseteq$ **PP.**
d. is juist, zie slides.

Vraag 6. Veronderstel, dat er een **P**-probleem wordt gevonden, dat tevens **PSPACE-volledig** blijkt te zijn. Welke van de volgende uitspraken is/zijn in dat geval waar?

a. **P** = **NP**.
b. **PSPACE** = **P**.
c. **NPC** $\subseteq$ **coNP**.
d. **coNP** = **NP**.

antwoord stel X $\in$ **P** $\cap$ **PSPACE-volledig**. Dan is ieder probleem in PSPACE reduceerbaar naar X. maar dan is ieder probleem in **NP, NPC** en **coNP** ook reduceerbaar naar X, derhalve zou gelden **PSPACE** $\subseteq$ **P, NP** $\subseteq$ **P**, **NPC** $\subseteq$ **P, coNP** $\subseteq$ **P.** Hieruit volgt:
a. is juist.
b. is juist.
c. is juist.
d. is juist.

Vraag 7. Gegeven zijn n processoren P_1 t/m P_n en een verzameling taken T. Van iedere taak $t \in T$ is de verwerkingstijd $v(t)$ in seconden bekend. Verder is er een deadline D (in gehele seconden) gegeven.
Gevraagd wordt of het mogelijk is de taken uit T zodanig over de n processoren te verdelen, dat elke processor P_i niet meer dan D seconden nodig zal hebben om alle aan P_i toebedeelde taken *achter elkaar* uit te voeren.

Dit probleem is:

a. een **NP-volledig** probleem, immers het is een restrictie van PARTITION.
b. een **NP-volledig** probleem: PARTITION is een restrictie van dit probleem.
c. een getalprobleem.
d. voor $n = 1$ een **P**-probleem en voor $n \geq 2$ een **NP-volledig** probleem.

antwoord a. is onjuist, PARTITION is juist een restrictie van dit probleem.
b. is juist, het is een **NP**-probleem en PARTITION is een restrictie van dit probleem.
c. is juist, verwerkingstijden en deadline zijn niet polynomiaal gebonden in de grootte van de instantie.
d. is juist, voor n=1 is er een triviaal polynomiaal algoritme (check of de verwerkingstijden sommeren tot een getal $\leq$ D), voor n=2 is PARTITION is nog steeds een restrictie van dit probleem.

Vraag 8. Het probleem om, gegeven een instantie (U,C) van SAT, te bepalen of er een waarheidstoekenning τ: $U \rightarrow \{0,1\}$ bestaat die tenminste de helft van het aantal clauses in C waarmaakt is een

a. **NP-volledig** probleem.
b. **coNP-volledig** probleem.
c. **P**-probleem.
d. probleem in **NP** $\cap$ **coNP**.

antwoord alleen alternatief c. en d. zijn juist.
Het antwoord op iedere instantie van zo'n probleem is namelijk altijd ja: een waarheidstoekenning die minstens de helft van het aantal clauses waarmaakt bestaat namelijk altijd.
Neem maar eens het volgende polynomiale algoritme:

while C $\neq \varnothing$, do
- neem een literal u $\in$ U die positief of negatief in enige clause c $\in$ C voorkomt.
- selecteer alle clauses c $\in$ C waarin u en/of $\neg$u voorkomt; en noem deze verzameling C_u ;
- kies die waarheidswaarde voor u die de meeste clauses uit C_u waarmaakt;
- verwijder vervolgens de verzameling C_u uit C;

De resulterende waarheidstoekenning zal altijd minstens de helft van alle clauses waarmaken.
Merk op, dat hiermee ook alternatief d. juist is omdat **P** $\subseteq$ **NP** $\cap$ **coNP**.

Vraag 9. Van een benaderingsalgoritme M voor een probleem A is bekend dat de performanceratio voor een instantie $x \in$ A gelijk is aan 4. Er is een andere instantie $y \in$ A is waarvoor de performanceratio gelijk is aan 2.
Wat kun je hieruit opmaken ten aanzien van de approximatieratio r_M van M?

a. $2 \leq r_M \leq 4$
b. $r_M \geq 4$.
c. $1 \leq r_M \leq 2$.
d. $1 \leq r_M \leq 4$.

antwoord de approximatieratio is de grootste waarde r_M waarvoor geldt dat r_M kleiner of gelijk is aan de performance ratio $R_M(x)$ voor alle instanties x van A. Uit de gegevens kun je nu alleen concluderen dat $r_M \geq 4$.
a. is onjuist
b. is juist.
c. is onjuist.
d. is onjuist.

Vraag 10. Een instantie van 5-SAT is een verzameling C van m clauses over een verzameling U van propositiesymbolen. Hierbij bevat iedere clause in C precies 5 literals.
Er geldt nu voor dit probleem dat:

a. een instantie die hoogstens 31 clauses bevat, altijd vervulbaar is.
b. er een probabilistisch algoritme bestaat voor het bepalen van het maximum aantal vervulbare clauses. Dit algoritme heeft een verwachte approximatieratio gelijk aan 32/31.
c. een instantie die hoogstens 32 clauses bevat, altijd vervulbaar is.
d. een probabilistisch algoritme bestaat dat met een kans hoogstens gelijk aan 1/32 een foute beslissing neemt.

antwoord alternatief a is juist. Een instantie van 5-SAT met minder dan 2^5 clauses is altijd vervulbaar (iedere clause sluit één waarheidstoekenning uit).
alternatief b. is juist. Een willekeurige waarheidstoekenning maakt een verwacht aantal gelijk aan m.31/32 clauses waar, waarbij m het aantal clauses is, als de set van clauses vervulbaar is.
alternatief c. is niet juist (zie a.)
alternatief d. is onjuist: Het is niet zo dat de kans $\leq$ 1/32 is dat een foute beslissing genomen wordt. Als dit zo zou zijn zouden we iha kunnen concluderen dat **BPP = NP.**

OPEN VRAGEN I (30 PUNTEN)

Onderstaande vragen dient u kort te beantwoorden.
Een antwoord dient in hoogstens 5-10 regels gegeven te worden.

Vraag 11. Uitgaande van het bekende NP-volledige CLIQUE-probleem wordt het volgende probleem X-CLIQUE geformuleerd:

Instantie een graaf $G = (V,E)$, een positieve integer K en twee gegeven knopen v en w in V;
Vraag is er een clique van G ter grootte van K die niet de knoop v bevat, maar wel de knoop w?

Is dit probleem een **coNP**-probleem, een **NP**-probleem of een probleem in **coNP**∩**NP**? Geef precies aan waarom.

antwoord Dit is een **NP**-probleem: Als certificaat dient een verzameling knopen V'. Test of V' een clique is, minstens K knopen bevat en ga na w een element van deze verzameling is, terwijl v niet voorkomt. Dit is in totaal in $O(|I|^2)$-tijd te verifiëren.
Het is niet bekend hoe nee-instanties in polynomiale tijd zouden kunnen worden geverifieerd.

Vraag 12. TIME(n^k) is de klasse van talen die in $O(n^k)$-tijd kunnen worden geaccepteerd door een DTm. Waarom geldt dat voor elke $k \geq 1$, TIME(n^k) strikt bevat is in TIME(n^{k+1})?

antwoord De tijd-hierarchiestelling zegt dat als $f(n)$ en $g(n)$ twee functies zijn zodat $\lim_{n\to\infty} f(n)/g(n) = 0$, dan is TIME($f(n)$) strict bevat in TIME($g(n)$.log $g(n)$).
Laat nu $f(n) = n^k$ **en** $g(n) = n^{k+1}/(k+1)\log n$ dan geldt $\lim_{n\to\infty} n^k (k+1)\log n / n^{k+1} = 0$.
Dus TIME(n^k) is strikt bevat in TIME(n^{k+1}) want
$\text{TIME}(g(n).\log g(n)) = \text{TIME}(n^{k+1}/(k+1)\log n \times \log (n^{k+1}/(k+1)\log n)) =$
$= \text{TIME}(n^{k+1}(1- \log(k+1)\log n/ (k+1).\log n) \subseteq \text{TIME}(n^{k+1})$

Vraag 13. Laat zien dat $\mathbf{P}^{\text{TQBF}} = \mathbf{NP}^{\text{TQBF}} = \mathbf{PSPACE}$.

antwoord Er geldt $\mathbf{P}^{\text{TQBF}} \subseteq \mathbf{NP}^{\text{TQBF}} \subseteq \mathbf{PSPACE}^{\mathbf{PSPACE}} = \mathbf{PSPACE}$.
Omdat TQBF een **PSPACE**-compleet probleem is en $\mathbf{P}^{\mathbf{PSPACE}} = \mathbf{PSPACE}$ geldt $\mathbf{P}^{\text{TQBF}} = \mathbf{PSPACE}$. Uit $\mathbf{PSPACE} \subseteq \mathbf{P}^{\text{TQBF}} \subseteq \mathbf{NP}^{\text{TQBF}} \subseteq \mathbf{PSPACE}$ volgt de gelijkheid.

Vraag 14. A en B zijn twee talen over een gegeven alfabet Σ.
Geef een eenvoudig voorbeeld om te laten zien dat uit $A \subseteq B$ niet altijd volgt $A \leq B$.

antwoord Neem een taal $A \neq \Sigma^*$ en laat $B = \Sigma^*$. Er geldt dan $A \subseteq B$. Echter, er is geen polynomiale reductie van A naar B mogelijk: nee-instanties $\Sigma^* - A$ kunnen niet afgebeeld worden op $\Sigma^* - B = \varnothing$.

Vraag 15. Waarom kan het algoritme van Shor niet gebruikt worden om aan te tonen dat alle problemen in **NP** met quantum computing in polynomiale tijd opgelost kunnen worden?

antwoord Het algoritme van Shor is een polynomiaal quantum algoritme voor het factorizeringsprobleem. Dit is een probleem in **NP** ∩ **coNP**. Deze klasse vormt, voorzover we weten, een strikt deelverzameling van **NP:** Geen enkel **NPC**-probleem behoort tot deze klasse en voor NPC problemen zijn dan ook nog geen polynomiale quantum algoritmen gevonden.

OPEN VRAGEN II (20 PUNTEN)

Beantwoord onderstaande vragen zo kort mogelijk. Geef geen irrelevante informatie, dit kan tot puntenaftrek aanleiding geven.

Vraag 16. (10 punten)

Iemand beweert dat LANGSTE PAD een **P**-probleem is. Zijn redenering is als volgt:

> Een instantie van LANGSTE PAD bestaat uit een graaf $G = (V,E)$ en een integer $K \geq 1$, en om na te gaan of G een simpel pad ter lengte van K bevat, kun je het volgende algoritme toepassen:
>
> 1. Selecteer alle deelverzamelingen E' uit E die precies K kanten bevatten;
> 2. Ga voor iedere deelverzameling E' na of uit de K kanten een simpel pad ter lengte van K te vormen is;
> 3. G is een yes-instantie als er minstens één E' te vinden is waaruit zo'n simpel pad ter lengte van K samen te stellen is; anders is G een no-instantie.
>
> Voor dit algoritme geldt:
>
> - Er zijn hoogstens $O(|E|^K)$ deelverzamelingen E' van E ter grootte van K.
> - Per deelverzameling E' kost het $O(K^2)$-tijd om na te gaan of er een simpel pad ter lengte van K te vormen is uit de kanten van E'.
> - Het totale algoritme kost derhalve $O(|E|^K \times K^2)$-tijd.
>
> Dit algoritme is derhalve polynomiaal in de grootte $|G| = |V| + |E|$ van een instantie en derhalve bestaat er een polynomiaal algoritme voor LANGSTE PAD.

Geef precies aan waarom deze redenering niet deugt.

antwoord De conclusie is fout: een algoritme met looptijd $O(|E|^K \times K^2)$ is niet polynomiaal in de grootte van de probleeminstantie. Strikt genomen is de grootte |I| van de probleem instantie gelijk aan $|I| = |V| + |E| + \log K$. Nemen we nu bv. $K = |V|/2 = n/2$, dan is de looptijd van het algoritme $O(|E|^{n/2} \times n^2)$. Voor instanties met $|E| = |V| = \Theta(n)$ geldt nu dat de grootte van de instantie gelijk is aan $\Theta(n)$, terwijl het algoritme $O(n^{n/2} \times n^2)$ run-time kost. Dit is niet polynomiaal in de grootte van de instantie. Merk tevens op dat hiermee het aantal deelverzamelingen E' ter grootte van K ook niet polynomiaal is in |I|.

Vraag 17. (10 punten)

2-TOURS is het volgende probleem

Instantie een verzameling S van n steden, een afstandenmatrix van afstanden $d_{ij} \in \mathbb{Z}^+$ tussen steden s_i en s_j in S en een positief geheel getal B.

Gevraagd bestaan er twee gescheiden tours langs steden in S zodat elke stad s in S deel uitmaakt van exact één tour en de totale afstand van de twee tours tezamen niet meer dan B bedraagt?

a. Laat zien dat 2-TOURS een **NP**-probleem is.

b. Geef een polynomiale-tijd reductie van TSP naar 2-TOURS.
Je kunt hierbij volstaan met de formulering van de reductie; een correctheidsbewijs is niet nodig.

antwoord

a. Neem twee rijtjes S1 en S2 van steden uit S. Ga na dat ze geen gemeenschappelijke elementen bevatten (kost $O(n^2)$) .
Ga na dat ieder element van S in een van de rijtjes voorkomt (kost $O(n^2)$)
Ga vervolgens na dat de som van de afstanden afgelegd in de tours S1 en S2 hoogstens gelijk is aan B. (kost $O(\Sigma \log d_{ij})$)

Het resultaat is een looptijd $O(n^2 + \Sigma \log d_{ij}) = O(|I|)$. Derhalve geldt dat een kandidaat oplossing in polynomiale tijd verifieerbaar is.

b. Neem het volgende algoritme:

input: een instantie I van TSP bestaande uit een verzameling van steden S , een afstandenmatrix $D = [d_{ij}]_{n \times n}$ en een bovengrens B.
output: een instantie $I' = (S', D', B')$ van 2-TOURS.

$S' := S \cup \{ s_{n+1}, s_{n+2} \}$; % $s_{n+1}, s_{n+2} \notin S$
$D' := [d'_{ij}]_{(n+2) \times (n+2)}$ waarbij $d'_{ij} = d_{ij}$ voor alle $1 \le i,j \le n$, $d'_{(n+1)j} = d'_{(n+2)j} = B+1$ voor alle $1 \le j \le$ n en $d'_{(n+1)j} = d'_{(n+2)j} = 0$ voor $j = n+1, n+2$;
$B' = B$.

Omdat er altijd een tour bestaat tussen de steden s_{n+1} en s_{n+2} met kosten 0 en geen enkele tour die een stad s_j bevat voor $j \le n$ een van de steden s_{n+1} of s_{n+2} kan bevatten (de totale kosten van een link tussen zulke steden is immers groter dan B) geldt nu dat de gecreeerde 2-TOURS instantie een yes-instantie is alss de oorspronkelijke TSP instantie een yes-instantie is.

EINDE TENTAMEN