Twaalfde college complexiteit

14/15 april 2008

NP-volledigheid III

Definitie.
Een algoritme heet polynomiaal begrensd als zijn worst case complexiteit van boven begrensd is door een functie die polynomiaal is in de lengte van de invoer.
Dus: worst case is $O(|x|^k)$ met $k \geq 0$ en met $|x|$ de lengte van de invoer x (of $O(n^k)$ met n een maat voor de lengte van de invoer).

Definitie
Een probleem heet polynomiaal begrensd als er een polynomiaal begrensd algoritme voor bestaat.

Definitie
De klasse van beslissingsproblemen die polynomiaal begrensd zijn noteren we als $\mathcal{P}$.

Een niet-deterministisch algoritme bestaat uit 3 fasen:

1. Niet-deterministische gokfase
   Er wordt een willekeurige string s in het geheugen geschreven. Elke keer dat het algoritme executeert kan dit een andere string zijn.
   // deze string is (hopelijk) het certificaat, het is een
   // soort gok van de oplossing van het probleem;
   // s kan echter ook een onzinstring zijn.

2. Deterministische verificatiefase
   Zowel de invoer van het probleem als de string s mogen hier gebruikt worden. Er wordt True of False geretourneerd of het programma gaat in een oneindige loop en stopt nooit.

// hier wordt gecontroleerd of s een oplossing van het
// probleem is bij de gegeven invoer, m.a.w. er wordt
// gecontroleerd of s een ja-antwoord rechtvaardigt.

3. Uitvoerstap
   Als fase 2 True retourneert geeft het algoritme als antwoord "ja". Anders is er geen uitvoer.

Het aantal stappen dat een niet-deterministisch algoritme doet is het aantal stappen nodig om s te schrijven (dus het aantal karakters waaruit s bestaat) + het aantal stappen dat gedaan wordt in de verificatiefase (+1 voor fase 3).

Een niet-deterministisch algoritme A antwoordt "ja" op invoer $x \Longleftrightarrow$ er bestaat een executie (dus een string s) van A op x die "ja" als uitvoer geeft. De invoer x is dan een ja-instantie. Als er geen ja-executie bestaat is het antwoord "nee" (x is dan een nee-instantie).

A is polynomiaal (begrensd) als er een ja-executie bestaat* die polynomiaal is in $|x|$. Dit is het geval als voor zo'n ja-executie $|s| = O(|x|^l)$ en Fase 2 polynomiaal is in $|x|$ en $|s|$ (dus $O(|x|^k \cdot |s|^m)$).

$\mathcal{NP}$ is de klasse van beslissingsproblemen waarvoor er een polynomiaal begrensd niet-deterministisch algoritme bestaat. $\mathcal{NP}$ betekent: $\mathcal{N}$on-deterministic $\mathcal{P}$olynomial time.

*die kan dus alleen bestaan voor ja-instanties

Stelling. Alle zes voorbeeldproblemen zitten in $\mathcal{NP}$.

Voorbeeld 1: Kleur
Gegeven een ongerichte graaf $\mathcal{G} = (V, E)$ en een geheel getal $k > 0$. Bestaat er een kleuring van $\mathcal{G}$ die precies k kleuren gebruikt (ofwel, is $\mathcal{G}$ k-kleurbaar)?
Probleeminvoer: $x =< \mathcal{G}, k >$.

Voorbeeld 2: HC
Gegeven een (gerichte of ongerichte) graaf $\mathcal{G}$. Vraag: heeft deze graaf een Hamiltonkring? Probleeminvoer: $x = \mathcal{G}$.
Laat hierna $V = \{1, 2, \ldots, n\}$ en dus $|V| = n$.

Voorbeeld 3: SAT
Gegeven een logische formule ϕ in CNF. Bestaat er een waardering van de in ϕ voorkomende logische variabelen die ϕ waar maakt? Probleeminvoer: $x = \phi$.

Een polynomiaal begrensd *niet-deterministisch algoritme* voor HC:

1. Fase 1 (gokfase)
   Er wordt een string s gegenereerd, hierna te interpreteren als een rij gehele getallen.

2. Fase 2 (verificatiefase)
   Er wordt gecontroleerd of s een Hamiltonkring voorstelt:

   (1) controleer dat er precies n integers staan: $O(|s|)$

   (2) controleer dat elke integer tussen 1 en n is: $O(|s|)$

   (3) controleer dat alle knopen uit s verschillend zijn: $O(|s|^2)$

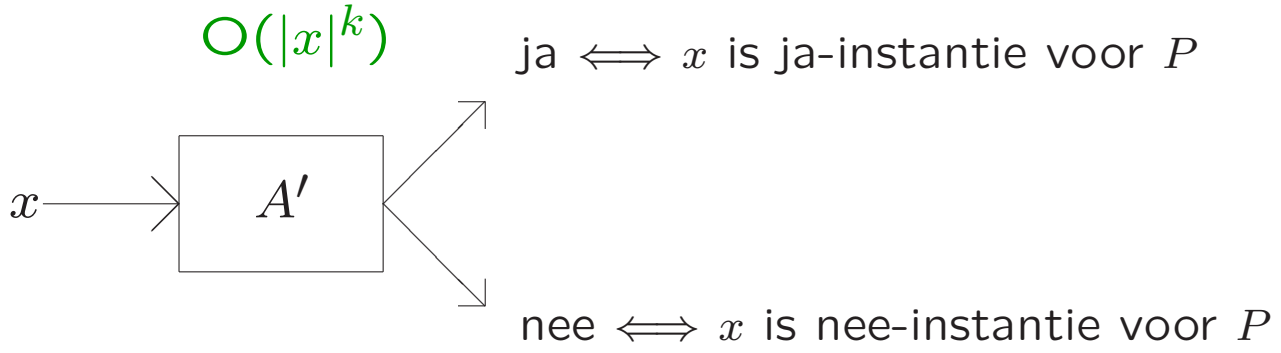
(4) controleer dat tussen opeenvolgende knopen uit s een tak zit in de graaf (en tussen de laatste en de eerste): $O(|s| \cdot |E|) = O(|s| \cdot |\mathcal{G}|)$.

Als de vier tests positief zijn wordt True geretourneerd, anders False (of er wordt in een oneindige loop gegaan).

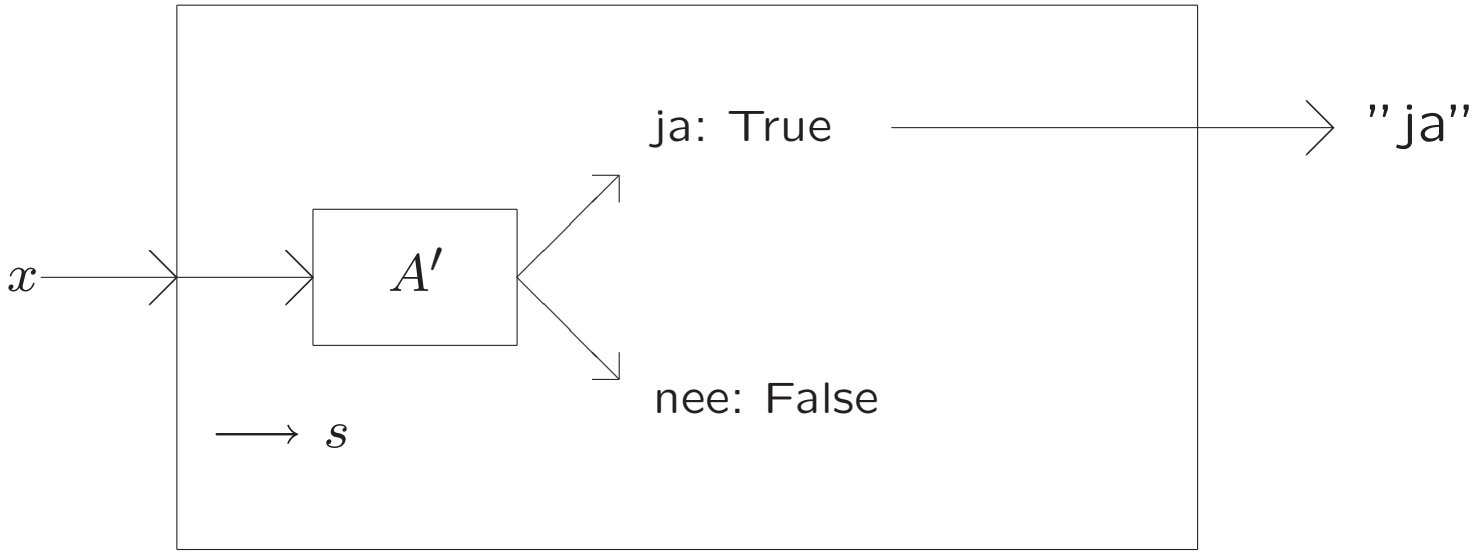
3. Fase 3 (uitvoerfase)
   Als fase 2 True oplevert wordt "ja" uitgevoerd, anders geen uitvoer.

Merk op:
Stap (1) t/m (3) controleert dat s een rij van n verschillende knopen van $\mathcal{G}$ voorstelt (soort syntactische controle), stap (4) controleert dat het een Hamiltonkring is.

Neem een willekeurig probleem $P \in \mathcal{P}$. Dan is er een polynomiaal deterministisch algoritme A' dat P oplost.



Het volgende niet-deterministische algoritme A is dan een polynomiaal begrensd algoritme voor P.



fase 1: genereer s;

fase 2: negeer de string s en voer A' uit op x;

fase 3:

Laat $V = \{1, 2, \ldots, n\}$, en de mogelijke kleuren $1, 2, \ldots, k$. Een polynomiaal begrensd *niet-deterministisch algoritme* voor Kleur is:

1. Fase 1 (gokfase)
   Er wordt een string s gegenereerd, hierna te interpreteren als een rij gehele getallen.

2. Fase 2 (verificatiefase)
   Er wordt gecontroleerd of s een goede kleuring voorstelt (s_i de kleur van knoop i):

   (1) controleer dat er precies n integers staan (elke knoop een kleur): kan p[olynomiaal, O($|s|$)

   (2) controleer dat elke integer tussen 1 en k is (er worden k kleuren gebruikt): kan polynomiaal, O($|s|$)

   (3) controleer dat aangrenzende knopen verschillend gekleurd zijn. Takken (v, w) aflopen en in s de kleur van v en w opzoeken en vergelijken: O($|E| \cdot |s|$) = O($|\mathcal{G}| \cdot |s|$).

Als de vier tests positief zijn wordt True geretourneerd, zodra een test negatief uitvalt wordt False teruggegeven (of er wordt in een oneindige loop gegaan).

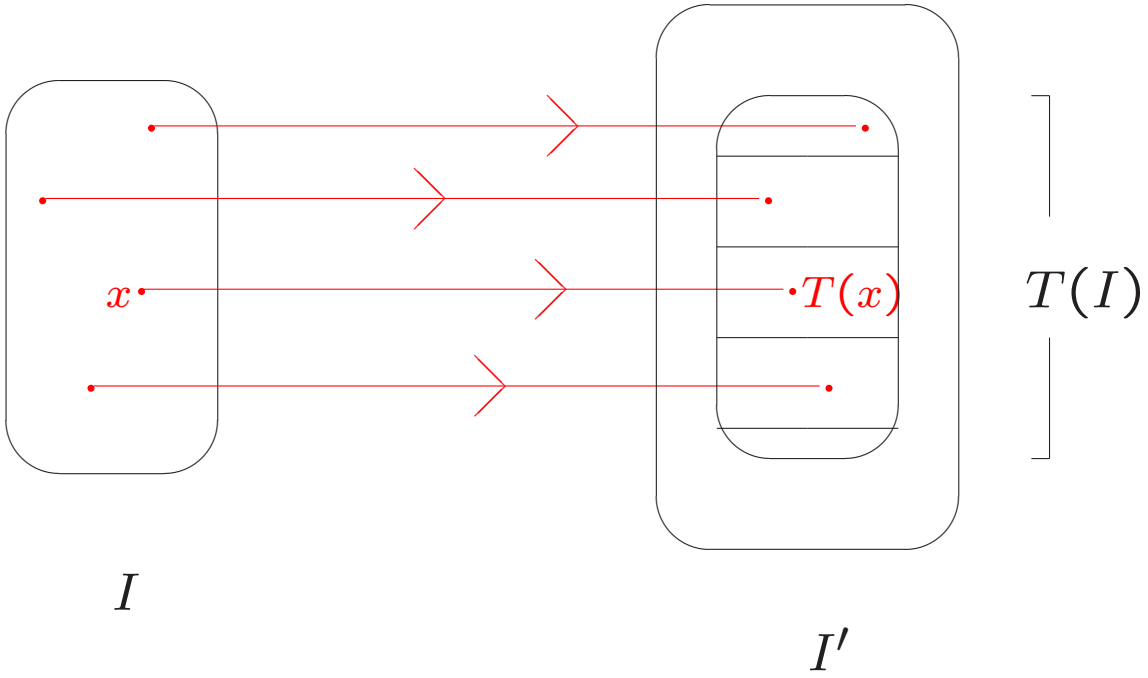
3. Fase 3 (uitvoerfase)
   Als fase 2 True oplevert wordt "ja" uitgevoerd, anders geen uitvoer.

Merk op. Stap (1) t/m (2) controleert dat s een kleuring van de knopen voorstelt stap (3) controleert dat het een correcte kleuring is.

Er geldt: $< \mathcal{G}, k >$ is een ja-instantie voor Kleur $\Longleftrightarrow$ er bestaat een goede kleuring van de knopen $\Longleftrightarrow$ er bestaat een goede string s waarop Fase 2 True oplevert $\Longleftrightarrow$ het antwoord van A op invoer $x =< \mathcal{G}, k >$ is "ja".

Verder: voor een ja-executie, dus met s een goede kleuring, is $|s| = \mathrm{O}(|V|) = \mathrm{O}(|x|)$. Ergo: A is polynomiaal begrensd.

Zij T een functie van de invoerverzameling I van een beslissingsprobleem P naar de invoerverzameling I' van een beslissingsprobleem Q.
T beeldt dus elke $x \in I$ af op een $T(x) \in I'$.



Definitie T heet een polynomiale reductie (of polynomiale transformatie) van P naar Q als de volgende drie punten gelden:

1. T kan berekend worden in polynomiaal begrensde tijd (als functie van $|x|$). D.w.z.: de constructie van $T(x)$ uit x kan in $O(|x|^k)$ stappen ($k \geq 0$).

2. Voor elke x uit I geldt: als x een ja-instantie is voor P dan is $T(x)$ een ja-instantie voor Q.

3. Voor elke x uit I geldt: als x een nee-instantie is voor P dan is $T(x)$ een nee-instantie voor Q.

3′. Voor elke x uit I geldt: als $T(x)$ een ja-instantie is voor Q dan is x een ja-instantie voor P. (Dit is equivalent met 3.)

Definitie

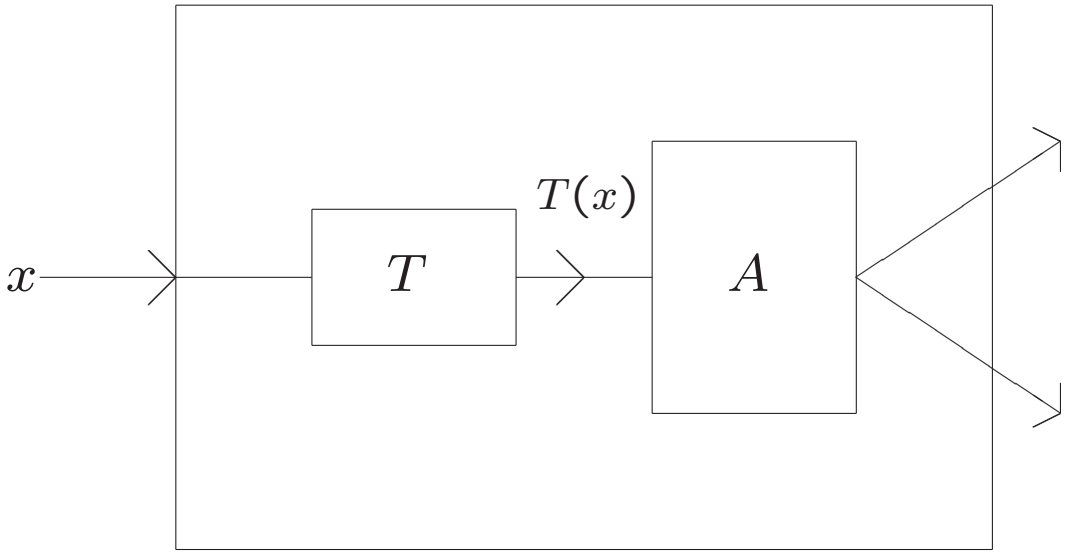
Een probleem P is polynomiaal reduceerbaar (of polynomiaal transformeerbaar) naar Q als er een polynomiale reductie bestaat van P naar Q.

Notatie: $P \leq_P Q$.

Stelling

Als $P \leq_P Q$ en Q zit in $\mathcal{P}$, dan zit P ook in $\mathcal{P}$.

Bewijs stelling



"ja" $\Longleftrightarrow$ $T(x)$ is ja-instantie van Q $\Longleftrightarrow$ x is ja-instantie van P

"nee" $\Longleftrightarrow$ $T(x)$ is nee-instantie van Q $\Longleftrightarrow$ x is nee-instantie van P

$O(|x|^k)$ $\qquad$ $O(|y|^l)$

$O(|x|^{kl})$

Definitie
Een probleem Q is NP-hard als elk probleem P in $\mathcal{NP}$ polynomiaal reduceerbaar is tot Q, dat wil dus zeggen dat $P \leq_P Q$ voor alle $P \in \mathcal{NP}$.

Definitie
Een probleem Q is NP-volledig als

1. $Q \in \mathcal{NP}$

2. Q is NP-hard

Notatie
De klasse van NP-volledige problemen geven we aan met $\mathcal{NPC}$ (NP-complete).

HP: **gegeven** een ongerichte graaf $\mathcal{G} = (V, E)$.
Vraag: heeft $\mathcal{G}$ een Hamiltonpad?

HC: **gegeven** een ongerichte graaf $\mathcal{G} = (V, E)$.
Vraag: heeft $\mathcal{G}$ een Hamiltonkring?

Transformeer een ongerichte graaf $\mathcal{G}$ (invoer voor HP) naar een ongerichte graaf $\mathcal{G}'$ (invoer voor HC) door één knoop toe te voegen en deze met een tak te verbinden met alle andere knopen. Noem deze transformatie T. Deze transformatie is een polynomiale reductie, dus er geldt (via deze T): HP $\leq_P$ HC.

HC1: **gegeven** een *gerichte* graaf $\mathcal{G}= (V, E)$.
Vraag: heeft $\mathcal{G}$ een Hamiltonkring?

HC2: **gegeven** een *ongerichte* graaf $\mathcal{G} = (V, E)$.
Vraag: heeft $\mathcal{G}$ een Hamiltonkring?

Bewering: HC1 $\leq_P$ HC2

Opmerking: er geldt ook: HC2 $\leq_P$ HC1. Bedenk zelf een eenvoudige reductie.

Transformatie T die een gerichte graaf $\mathcal{G} = (V, E)$ op een ongerichte graaf $T(\mathcal{G}) = \mathcal{G}' = (V', E')$ afbeeldt:

- $V' = \{v_1, v_2, v_3 : v \in V\}$: elke knoop $v \in V$ wordt afgebeeld op een drietal knopen v_1, v_2, v_3.
- $E' = \{(v_1, v_2), (v_2, v_3) : v \in V\} \cup \{(v_3, w_1) : (v, w) \in E\}$: binnen elk drietal knopen corresponderend met v loopt een tak tussen v_1 en v_2 en tussen v_2 en v_3, en voor elke tak van v naar w in $\mathcal{G}$ komt een tak in $\mathcal{G}'$ tussen v_3 en w_1.

Dan geldt:

1. T kan in polynomiaal begrensde tijd berekend worden (constructie van $T(\mathcal{G})$ uit $\mathcal{G}$ is $O(|\mathcal{G}|)$)
2. $\mathcal{G}$ is een ja-instantie voor HC1 $\iff$ $\mathcal{G}'$ is een ja-instantie voor HC2, ofwel: $\mathcal{G}$ heeft een gerichte Hamiltonkring $\iff$ $\mathcal{G}'$ heeft een ongerichte Hamiltonkring

Laatste week

- maandag 21 april: 11.15-13.00 college
- maandag 21 april: 13.45-15.30 laatste werkcollege
- dinsdag 22 april: 11.15-13.00 laatste college
- dinsdag 6 mei: 11.15-13.00 KI, dus geen wCom
- tentamen: donderdag 5 juni 2008, 14.00-17.00
- vragenuur: ?????