Uitwerking opgaven werkcollege 12

Opgave 68

a. We mogen wel aannemen dat het aantal knopen bekend is en gelijk aan n, dat $V = \{1, \ldots, n\}$ en dat $\mathcal{G}$ met een adjacencylist wordt gerepresenteerd. Idee: gok een string s die hopelijk een correcte onafhankelijke knoopverzameling voorstelt en controleer dat dat klopt. Als die controle polynomiaal is in $|x| = |\langle G, K\rangle|$, zit InSet in $\mathcal{NP}$.

Een niet-deterministisch algoritme A voor InSet:

- Fase 1 (gokfase)
  Er wordt een string s gegenereerd, te intepreteren als een rij getallen.
- Fase 2 (verificatiefase) Er wordt gecontroleerd of s een onafhankelijke verzameling in $\mathcal{G}$ voorstelt:
  - controleer dat elk getal s_i uit s tussen 1 en $|V|$ zit: s aflopen en vergelijken, dus $O(|s|)$.
  - controleer dat elke s_i precies één keer in s voorkomt: $O(|s|^2)$.
  - controleer dat s precies K waarden bevat: s aflopen, het aantal elementen tellen en vergelijken met K, dus $O(|s|)$.

    Als aan deze drie voorwaarden voldaan is stelt s inderdaad een deelverzameling van de knopen voor ter grootte K. Nu nog controleren dat s een onafhankelijke verzameling is. Als dit niet het geval is wordt gestopt en False geretourneerd, of er wordt in een oneindige loop gegaan of ...

  - controleer dat voor elke twee waarden s_i en s_j $(i \neq j)$, gezien als knopen in V, geldt dat er geen tak is tussen s_i en s_j: alle tweetallen knopen uit s aflopen ($O(|s|^2)$) en voor elk tweetal kijken of de een in de buurlijst van de ander voorkomt, dus $O(|s|^2 \cdot |E|) \subseteq O(|s|^2 \cdot |x|)$.

  Als alle vier controles positief zijn retourneert het verificatie-algoritme True, anders wordt (zodra een test niet klopt) False geretourneerd of gaat het programma in een oneindige loop (of zo).
  Fase 2 geeft dus True d.e.s.d.a. s een onafhankelijke verzameling bestaande uit K knopen voorstelt.
- Fase 3 (uitvoerfase)
  Als Fase 2 True oplevert wordt "ja" uitgevoerd, anders geen uitvoer.

Nu geldt: het antwoord van het algoritme is "ja" dan en slechts dan als er een executie van A bestaat die "ja" oplevert d.e.s.d.a. er een string s bestaat waarop A in Fase 2 True oplevert d.e.s.d.a. er een string s bestaat die een onafhankelijke verzameling voorstelt van precies K knopen d.e.s.d.a. $\mathcal{G}$ een onafhankelijke

knoopverzameling heeft ter grootte K, dan en slechts dan als $\langle \mathcal{G}, K \rangle$ een ja-instantie is van InSet.

Het algoritme A geeft derhalve voor elke invoer x het juiste antwoord.

Verder is het algoritme polynomiaal. Namelijk: voor een ja-executie (dus met x een ja-instantie, en s een goede string) stelt s een deelverzameling van de knopen voor, dus dan is $|s| \leq |V| \leq |x|$. Derhalve is Fase 1 dan $O(|x|)$ en de verificatiefase $O(|x|^3)$. Dus voor elke invoer x waarvoor het antwoord van A "ja" is, is er een executie die dat in $O(|x|)$ (Fase 1) + $O(|x|^3)$ (Fase 2) $\subseteq O(|x|^3)$ stappen doet, in totaal polynomiaal in x.

Voor een ja-instantie geldt dat $|s| \in O(|G|)$, dus dan is alles polynomiaal in $|x|$ (met $x = \langle G, K \rangle$).

b.

1. Nee. Als P reduceert naar InSet, wil dat niet zeggen dat alles in $\mathcal{NP}$ ook reduceert naar P. Je weet alleen dat InSet minstens net zo moeilijk is als P. Het probleem P zou nog best in $\mathcal{P}$ kunnen zitten.
2. Ja. Als InSet reduceert naar P en InSet is NP-volledig (en dus NP-hard), dan reduceert alles in $\mathcal{NP}$ vanwege de transitiviteit van $\leq_{\mathrm{P}}$ naar P (stelling college). Dus P is NP-hard. Verder weten we dat $P \in \mathcal{NP}$, dus P is NP-volledig (definitie).
3. Nee. Uit de definitie van NP-volledigheid weten we dat alle problemen uit $\mathcal{NP}$ polynomiaal reduceren naar InSet. Echter weten we niet of $P \in \mathcal{NP}$; P zou bijvoorbeeld in $\mathcal{EXP}$ kunnen zitten.

c.

(i) Het probleem 2InSet is polynomiaal oplosbaar (deterministisch dus). De vraag is of een gegeven ongerichte graaf $\mathcal{G} = (V, E)$ een onafhankelijke knoopverzameling bevat die uit twee knopen bestaat, dus of er een tweetal knopen in $\mathcal{G}$ bestaat dat niet verbonden is[1]. Een polynomiaal algoritme daarvoor is: loop alle tweetallen knopen af: $O(|V|^2)$ stuks. Controleer van elk paar of er een tak tussen zit: $O(|E|)$ per tak ($O(|1)$ voor adjacencymatrix). Samen is dit $O(|V|^2 \cdot |E|) \subseteq O(|x|^3)$, dus polynomiaal.
We weten nu dat 2InSet $\in \mathcal{P}$. Als 2InSet NP-volledig is, zouden we een NP-volledig probleem hebben dat in $\mathcal{P}$ zit. Hieruit zou volgen dat $\mathcal{P} = \mathcal{NP}$ (zie college). Dit is hoogstwaarschijnlijk niet het geval, dus hoogstwaarschijnlijk bestaat er niet zo'n reductie.

(ii) Merk op dat 3InSet, 4InSet, ... stuk voor stuk polynomiaal oplosbaar zijn volgens eenzelfde brute force benadering als voor 2InSet. Voor – bijvoorbeeld– 4InSet genereren we alle mogelijke deelverzamelingen bestaande uit 4 knopen; dit zijn er $\binom{|V|}{4}$, en dat zijn er $O(|V|^4)$. Van elk daarvan controleren we of deze onafhankelijk is, dus lopen we alle tweetallen knopen uit elk viertal af (dat zijn er $\binom{4}{2} = 6$ per viertal) en kijken

[1] Een graaf is dus een ja-instantie van 2InSet dan en slechts dan als de graaf niet volledig verbonden is.

of er een tak tussen zit: $O(|E|)$ per tweetal, dus ook $O(|E|)$ per deelverzameling. Samen is dit dus $O(|x|^5)$. Analoog alle andere gevallem met vaste grootte van de onafhankelijke knoopverzameling.
Elk probleem jInSet is dus polynomiaal (deterministisch) oplosbaar (in $O(|x|^{j+1}$, dus wel onpraktisch voor grote j). Derhalve behoren ze allemaal (hoogstwaarschijnlijk) niet tot $\mathcal{NPC}$.
Als het aantal knopen in de onafhankelijke deelverzameling een parameter van het probleem wordt, dus deel uitmaakt van de invoer (dan hebben we het over het probleem InSet), is het probleem wel NP-volledig.

Opgave 4, tentamen 10/06/2016

Zie uitwerking website

Opgave 4, tentamen 08/07/2016

Zie (schets van de) uitwerking op de website