Complexiteit

College 7

Docent: Lieuwe Vinkhuijzen

9 maart 2020

Nieuws

- **Huiswerkopgave 2:**
  `liacs.leidenuniv.nl/~vinkhuijzenlt/complexiteit`
- **Deadline:** 30 maart 23:59
- Engelse huiswerkopgaven
- English exam: on request, send an email to
  `vinkhuijzenlt@liacs.leidenuniv.nl`

Vorige week

- Circuit Gelijkheid oplossen met circuit SAT
- Circuit SAT oplossen met CNF-SAT
- Getallen vermenigvuldigen in $\mathcal{O}(n^{\log_2(3)}) \approx \mathcal{O}(n^{1.58})$

Deze week

1. Reducties

CNF-Satisfiability oplossen met Circuit Satisfiability

Zij $T_{\text{Circuit}}(n)$ de tijdscomplexiteit van een algoritme voor Circuit Satisfiability.
Dan is CNF-Satisfiabiltiy ook in $T_{\text{Circuit}}(n)$ tijd op te lossen.

Zij $T_{\text{CNF}}(n)$ de tijdscomplexiteit van een algortime voor CNF-Satisfiability.
Dan is Circuit-Satisfiabiltiy in $T_{\text{CNF}}(cn)$ op te lossen, voor een zekere constante c.

Voorbeeld. Als CNF-Satisfiability op te lossen is in $\mathcal{O}(2^{0.1n})$ tijd, dan is Circuit satisfiability op te lossen in $\mathcal{O}(2^{0.1cn})$ tijd.
Gevolg. CNF-Satisfiability is in polynomiale tijd op te lossen dan en slechts dan als Circuit Satisfiability in polynomiale tijd op te lossen is.

Reducties

- Beslissingsproblemen (ja/nee)
- Constructieproblemen (sorteer deze lijst / vindt een x zdd)
- Optimaliseringsproblemen (vind de grootste x zdd)
- ...

Reducties

- **Beslissingsproblemen (ja/nee)**
- Optimaliseringsproblemen (vind de grootste x zdd)
- Constructieproblemen (sorteer deze lijst / vindt een x zdd)
- ...

Reducties: Beslissingsproblemen

Een tupel $L = (L_{\text{ja}}, L_{nee})$ heet een *beslissingsprobleem*.
L_{ja} en L_{nee} zijn verzamelingen. L_{ja} zijn de "ja-instanties" van het probleem, L_{nee} de "nee-instanties". Er geldt $L_{ja} \cap L_{nee} = \emptyset$.
Een algoritme A lost L op als, voor alle x geldt dat

1. $A(x) = \textsc{True}$ d.e.s.d.a. $x \in L_{ja}$
2. $A(x) = \textsc{False}$ d.e.s.d.a. $x \in L_{nee}$

Aanname vandaag: $L_{ja} \cup L_{nee} = \{0,1\}^*$.
Al gezien in FI1, FI2, FI3.

Reducties: Reducties

Definition (Reductie)

Zij K, L beslissingsproblemen. Een *reductie* van K naar L is een functie f die inputs $x \in K_{ja} \cup K_{nee}$ neemt, en een output $y = f(x) \in L_{ja} \cup L_{nee}$ geeft, zodanig dat

1. Als $x \in K_{ja}$ dan $f(x) \in L_{ja}$
2. Als $x \in K_{nee}$ dan $f(g) \in L_{nee}$

Schrijf $K \leq L$

Definition (Polynomiale tijd reductie)

f is een *polynomiale tijd reductie* als f een reductie is, en het algoritme dat f berekent heeft polynomiale tijdscomplexiteit.

Schrijf: $K \leq_P L$.

Reducties: Aanleiding

Theorem

Als $K \leq_P L$, en als er een algoritme is dat L oplost in tijd $T_L(|x|)$, dan is er polynoom p, en een algoritme A, z.d.d. A het probleem K oplost in tijd $T_L(p(|x|))$.

Corollary

Als $K \leq_P L$, dan is er een polynomiaal algoritme voor K als er een polynomiaal algoritme is voor K.

Corollary

Als $K \leq_P L$ en $L \leq_P K$ dan is er een polynomiaal algoritme voor beide, of voor geen van beide.

Ezelsbruggetje $\leq_P$

$5 \leq 10$	5 is niet groter dan 10
$\{2,3\} \subseteq \{2,3,5,7\}$	$\{2,3\}$ is niet groter dan $\{2,3,5,7\}$
$K \leq_P L$	K is niet moeilijker dan L

Circuit Satisfiability oplossen met CNF Satisfiability

Input: Een Bools $\{\wedge, \vee, \neg\}$-circuit C met inputs $x_1, \ldots, x_n$ en één output.
Output: "Ja" d.e.s.d.a. er een toekenning aan $x_1, \ldots, x_n$ bestaat die het circuit $\textsc{True}$ doet outputten.

Plan. We maken een CNF formule ϕ die *satisfiable* is d.e.s.d.a. C satisfiable is.
We "encoderen" de goede toekenningen aan C, in ϕ.
Zeg dat C, k gates heeft, $g_1, \ldots, g_k$. De formule ϕ heeft dan $n + k$ variabelen, namelijk $x_1, \ldots, x_n, g_1, \ldots, g_k$.

Circuit Satisfiability oplossen met CNF Satisfiability

Input: Een Bools $\{\wedge, \vee, \neg\}$-circuit C met inputs $x_1, \ldots, x_n$ en één output.
Output: "Ja" d.e.s.d.a. er een toekenning aan $x_1, \ldots, x_n$ bestaat die het circuit $\textsc{True}$ doet outputten.

$$\phi = (g_6) \wedge (g_6 = (g_4 \vee g_5)) \wedge (g_5 = (g_1 \wedge g_3)) \wedge (g_4 = (x_1 \wedge g_2))$$
$$\wedge (g_2 = (x_2 \vee x_3)) \wedge (g_3 = (x_3 \wedge x_3)) \wedge (g_1 = \neg x_1)$$

Staat nog niet in CNF, maar gaat de goede kant op.

Lemma

C is satisfiable d.e.s.d.a. ϕ satisfiable is.

Circuit Satisfiability oplossen met CNF Satisfiability

Input: Een Bools $\{\wedge, \vee, \neg\}$-circuit C met inputs $x_1, \ldots, x_n$ en één output.
Output: "Ja" d.e.s.d.a. er een toekenning aan $x_1, \ldots, x_n$ bestaat die het circuit $\textsc{True}$ doet outputten.

$$\phi = (g_6) \wedge (g_6 = (g_4 \vee g_5)) \wedge (g_5 = (g_1 \wedge g_3)) \wedge (g_4 = (x_1 \wedge g_2))$$
$$\wedge (g_2 = (x_2 \vee x_3)) \wedge (g_3 = (x_3 \wedge x_3)) \wedge (g_1 = \neg x_1)$$

Staat nog niet in CNF, maar gaat de goede kant op.

Lemma

C is satisfiable d.e.s.d.a. ϕ satisfiable is.

Naar CNF met DeMorgan's wetten (op het bord)

Perfect matching oplossen met SAT

Input: Een graaf $G = (V, E)$
Output: Een verzameling $M \subseteq E$, zodanig dat voor elke knoop $v \in V$: v raakt precies één tak in M.

Perfect matching oplossen met SAT

Input: Een graaf $G = (V, E)$
Output: Een verzameling $M \subseteq E$, zodanig dat voor elke knoop $v \in V$: v raakt precies één tak in M.

Plan. We maken een CNF formule, ϕ, die *satisfiable* is dan een slechts dan als G een perfect matching heeft.

1. We maken voor elke tak $e \in E$ een variabele x_e
2. We zorgen dat de satisfying assignments "precies overeenkomen met" de perfect matchings van G
3. We maken twee formules α en β, en dan zetten we $\phi = \alpha \wedge \beta$
4. α is een formule die satisfiable is d.e.s.d.a. elke knoop met een tak verbonden is
5. β is een formule die satisfiable is d.e.s.d.a. elke knoop met minder dan twee takken verbonden is

Perfect matching oplossen met SAT: Constructie van α

Input: Een graaf $G = (V, E)$
Output: Een verzameling $M \subseteq E$, zodanig dat voor elke knoop $v \in V$: v raakt precies één tak in M.

(Voorbeeld op het bord)

$$
\begin{aligned}
\alpha = & (x_{1,2} \vee x_{1,5}) && \text{Vertex 1} \\
& \wedge (x_{1,2} \vee x_{2,3} \vee x_{2,6}) && \text{Vertex 2} \\
& \wedge (x_{2,3} \vee x_{3,4} \vee x_{3,6}) && \text{Vertex 3} \\
& \wedge (x_{3,4} \vee x_{4,5} \vee x_{4,6}) && \text{Vertex 4} \\
& \wedge (x_{1,5} \vee x_{4,5} \vee x_{5,6}) && \text{Vertex 5} \\
& \wedge (x_{2,6} \vee x_{3,6} \vee x_{4,6} \vee x_{5,6}) && \text{Vertex 6}
\end{aligned}
$$

Reductie: Voorbeeld, Hamiltonkring

Definition (Hamiltonkring)

Een Hamiltonkring in een graaf is een cykel die alle knopen precies één keer bezoekt.

Probleem *GHK* (Gerichte Hamiltonkring)
Input: Een gerichte graaf
Output: **Ja** als G een Hamiltonkring heeft, anders **Nee**.

Probleem *OHK* (Ongerichte Hamiltonkring)
Input: Een ongerichte graaf
Output: **Ja** als G een Hamiltonkring heeft, anders **Nee**.

Reductie: Voorbeeld, Hamiltonkring

Probleem *GHK* (Gerichte Hamiltonkring)
Input: Een gerichte graaf
Output: **Ja** als G een Hamiltonkring heeft, anders **Nee**.

Probleem *OHK* (Ongerichte Hamiltonkring)
Input: Een ongerichte graaf
Output: **Ja** als G een Hamiltonkring heeft, anders **Nee**.

Reductie: Een transformatie f die een gerichte graaf $G = (V, E)$ afbeeldt op een nieuwe, ongerichte graaf $f(G) = G' = (V', E')$:

- $V' = \{v_1, v_2, v_3 | v \in V\}$
- $E' = \{(s_3, t_1) | (s, t) \in E\} \cup \{(v_1, v_2), (v_2, v_3) | v \in V\}$

Reductie: Voorbeeld, Hamiltonkring

Voldoet f aan de definitie van een reductie?

1. Als $G \in GHK$ dan $f(G) \in OHK$
2. Als $G \notin GHK$ dan $f(G) \notin OHK$

(Dus antwoord is: Ja, f is inderdaad een reductie)

---

3. f wordt berekend in polynomiaal veel tijd

1972: R. Karp: *Reducibility Among Combinatorial Problems*

1. Satisfiability
2. 3-CNF Satisfiability
3. 0/1 integer programming
4. Kliek
5. Set packing
6. Vertex cover
7. Set covering
8. Feedback node set
9. Hamilton Cykel
10. Graafkleuring
11. Clique cover
12. 3d matching
13. Knapzak
14. Job sequencing
15. Partition
16. Max cut

P, EXP, NP

1. **P** Problemen die polynomiaal oplosbaar zijn, $\mathcal{O}(|x|^k)$
2. **EXP** Problemen die in exponentieel tijd oplosbaar zijn, $\mathcal{O}(2^{|x|^k})$

P, EXP, NP

1. **P** Problemen die polynomiaal oplosbaar zijn, $\mathcal{O}(|x|^k)$
2. **EXP** Problemen die in exponentieel tijd oplosbaar zijn, $\mathcal{O}(2^{|x|^k})$
3. **NP** Problemen wiens oplossing snel te verifiëren is, mits hij bestaat

$$\mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{EXP} \tag{1}$$

P, EXP, NP

Definition (**P** (Polynomiale tijd))

Een taal $L \subseteq \{0,1\}^*$ is in **P** als er een $\mathcal{O}(|x|^k)$ tijd algoritme A bestaat dat het beslissingsprobleem $(L, \overline{L})$ oplost:

1. Als $x \in L$ dan $A(x) = 1$
2. Als $x \notin L$ dan $A(x) = 0$

Definition (**NP** (Non-deterministisch polynomiale tijd))

Een taal $L \subseteq \{0,1\}^*$ is in **NP** als er een $\mathcal{O}(|x|^k)$ tijd algoritme A bestaat, dat twee inputs x, y heeft, en er geldt

1. Als $x \in L$ dan $\exists y \colon A(x,y) = 1$
2. Als $x \notin L$ dan $\forall y \colon A(x,y) = 0$