Complexiteit

College 8

Docent: Lieuwe Vinkhuijzen

2 april 2020

Nieuws

- **Huiswerkopgave 2:**
  `liacs.leidenuniv.nl/~vinkhuijzenlt/complexiteit`
- **Nieuwe deadline:** 20 april

- Skype: lieuwe.vinkhuijzen1
- Vragen: `complexiteitleiden@gmail.com`

Vorige keer

- Reducties
- P, NP, EXP
- NP-Hardness

Deze week

1. NP-Hard, NP-Volledig
2. Cook-Levin Stelling: NP-Volledige problemen bestaan
3. Algoritme van Savitch

Klassificatie van talen tot nu toe

Taal: Een verzameling $L \subseteq \{0,1\}^*$.

$$\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{EXP}$$

Klassificatie van talen tot nu toe

Taal: Een verzameling $L \subseteq \{0,1\}^*$.

$$\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{EXP} \subsetneq \underbrace{\mathrm{R} \subsetneq \mathrm{RE}}_{\text{FI 3}}$$

Klassificatie van talen tot nu toe

Taal: Een verzameling $L \subseteq \{0,1\}^*$.

$$\underbrace{\textsc{Regular} \subsetneq \text{CFL}}_{\text{FI } 1+2} \subsetneq \text{P} \subseteq \text{NP} \subseteq \text{EXP} \subsetneq \underbrace{\text{R} \subsetneq \text{RE}}_{\text{FI } 3}$$

Recap

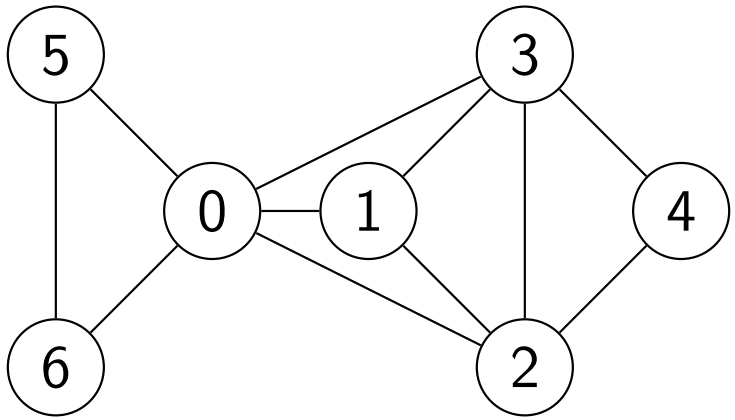
Taal: Een verzameling $L \subseteq \{0,1\}^*$.
Reductie: $L \leq_p K$, betekent L is hooguit zo moeilijk als K.
P: De verzameling problemen die in $\mathcal{O}(n^c)$ tijd beslist kunnen worden.
NP: De verzameling problemen waarvan een ja-instantie in $\mathcal{O}(n^c)$ herkend kan worden aan een certificaat.
NP-Hard: Als alle NP problemen reduceren naar L, is L NP-Hard.
NP-Volledig: Als L NP-Hard is, én $L \in \mathrm{NP}$.

Recap

Taal: Een verzameling $L \subseteq \{0,1\}^*$.
Reductie: $L \leq_p K$, betekent L is hooguit zo moeilijk als K.
P: De verzameling problemen die in $\mathcal{O}(n^c)$ tijd beslist kunnen worden.
NP: De verzameling problemen waarvan een ja-instantie in $\mathcal{O}(n^c)$ herkend kan worden aan een certificaat.
NP-Hard: Als alle NP problemen reduceren naar L, is L NP-Hard.
NP-Volledig: Als L NP-Hard is, én $L \in \mathrm{NP}$.
Cook-Levin Stelling: Er bestaat een NP-Volledig probleem, namelijk $\textsc{Satisfiability}$.

Kliek $\leq_{\mathrm{P}}$ SAT

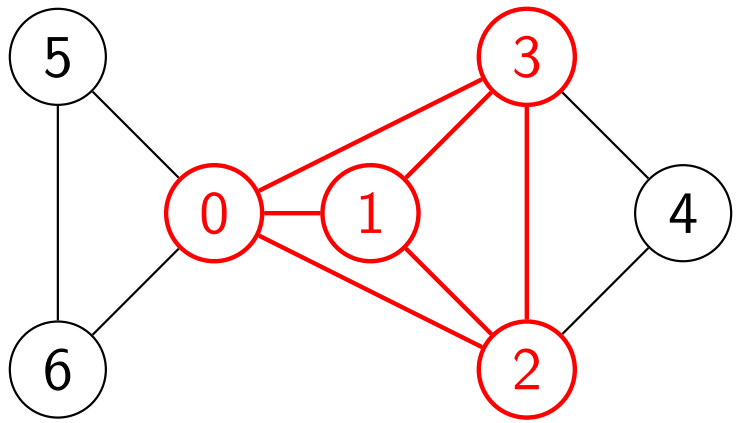
Input: Een ongerichte graaf $G = (V, E)$ en een getal $k \geq 0$.
Output: "*Ja*" desda G een kliek heeft ter grootte k.



Figuur: Een graaf met een kliek ter grootte 4

Kliek $\leq_{\mathrm{P}}$ SAT

Input: Een ongerichte graaf $G = (V, E)$ en een getal $k \geq 0$.
Output: "*Ja*" desda G een kliek heeft ter grootte k.



Figuur: Een graaf met een kliek ter grootte 4

Kliek $\leq_{\mathrm{P}}$ SAT

Input: Een ongerichte graaf $G = (V, E)$ en een getal $k \geq 0$.
Output: "*Ja*" desda G een kliek heeft ter grootte k.

We geven een reductie Kliek $\leq_{\mathrm{P}}$ SAT.
Plan:

1. Encodeer een verzameling knopen als een toekenning aan kn variabelen
2. We maken een formule ϕ die vervulbaar is desda G een kliek heeft van grootte k
   2.1 De vervullende toekenningen van ϕ zijn de klieks in G van k knopen

Kliek $\leq_{\mathrm{P}}$ SAT

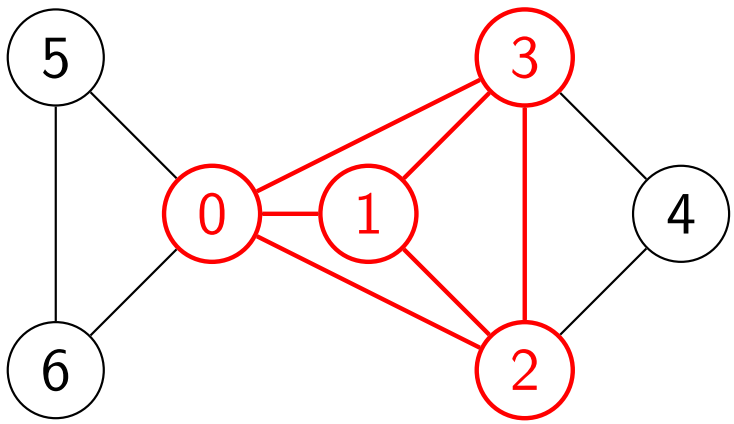
De formule ϕ heeft kn variabelen

$$\phi(x_0, \ldots, x_{n-1}, x_n, \ldots, x_{2n-1}, x_{2n}, \ldots, x_{kn-1})$$

Voorbeeld 1:

$\vec{x} = \underbrace{1000000}_{0}\ \underbrace{0100000}_{1}\ \underbrace{0010000}_{2}\ \underbrace{0001000}_{3}$

$\vec{x} \sim \{0, 1, 2, 3\}$

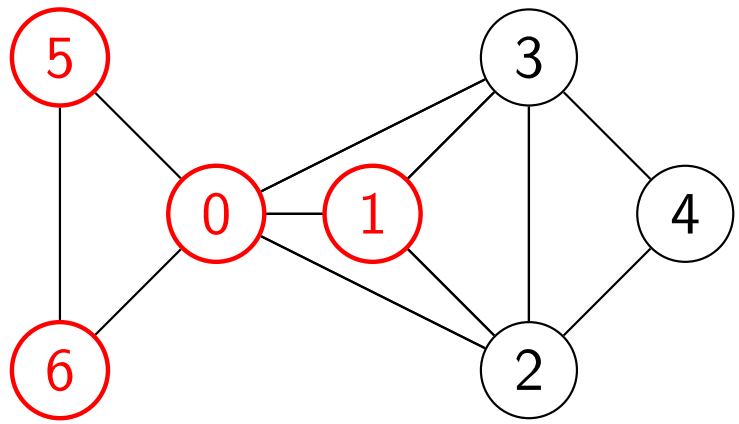


(a) Een graaf met een kliek ter grootte 4

Voorbeeld 2:

$\vec{x} = \underbrace{1000000}_{0}\ \underbrace{0100000}_{1}\ \underbrace{0000010}_{5}\ \underbrace{0000001}_{6}$

$\vec{x} \sim \{0, 1, 5, 6\}$



(b) Een graaf met een kliek ter grootte 4

Kliek $\leq_{\mathrm{P}}$ SAT

Input: Een ongerichte graaf $G = (V, E)$ en een getal $k \geq 0$.
Output: "*Ja*" desda G een kliek heeft ter grootte k.

De formule ϕ is een conjunctie (AND) van de volgende eisen:

1. Elk blok $x_{in}, \ldots, x_{i(n+1)-1}$ is één knoop
2. Alle knopen zijn verschillend
3. Tussen elke twee knopen is een tak in G

Kliek $\leq_{\mathrm{P}}$ SAT

Input: Een ongerichte graaf $G = (V, E)$ en een getal $k \geq 0$.
Output: "*Ja*" desda G een kliek heeft ter grootte k.

De formule ϕ is een conjunctie van de volgende clausulen:

1. $(x_{in} + x_{in+1} + \cdots + x_{i(n+1)-1} = 1)$ voor $0 \leq i < k$
   Elk blok is één knoop
2. $(\neg x_{in+v} \vee \neg x_{jn+v})$ voor $0 \leq i < j < k$ en $0 \leq v < n$
   Alle knopen zijn verschillend
3. $(\neg x_{in+u} \vee \neg x_{jn+v})$ voor $0 \leq i < j < k$ en $(u, v) \notin E$
   Als er geen tak (u, v) is, dan zijn u en v niet allebei $\textsc{True}$.

Cook-Levin: SAT is NP-Volledig

Zij $L \in \mathrm{NP}$ een taal in NP. We geven een reductie $L \leq_P \mathrm{SAT}$.

Plan:

1. Er is een non-deterministische Turing Machine T die L accepteert, en T draait voor hooguit $N(|x|) = |x|^c$ stappen.
2. Voor een input x maken we een formule ϕ, die vervulbaar is desda T de input x accepteert
3. Encodeer $N + 1$ "snapshots / configuraties" van de Turing Machine, dus we encoderen een berekening van T op x
   3.1 Een configuratie is een tripel (Tape, q, j) met tape inhoud, een toestand $q \in Q$, en positie op de tape $j \in \mathbb{Z}$
4. (De formule ϕ is lang, maar "slechts" polynomiaal lang in $|x|$)

Non-deterministische Turing Machines

Een Non-deterministische Turing Machine $T = (Q, \Sigma, \Gamma, q_0, \delta)$ bestaat uit:

- Σ: het leesalfabet. Bijvoorbeeld $\Sigma = \{0, 1\}$
- Γ: het schrijfalfabet; bevat niet Δ. Bijvoorbeeld $\Gamma = \{0, 1\}$.
- Q: een eindige verzameling toestanden, met begintoestand $q_0 \in Q$; met daarnaast q_a, q_r
- $\delta \subseteq (Q \setminus \{q_a, q_r\}) \times (\Gamma \cup \{\Delta\}) \times Q \times (\Gamma \cup \{\Delta\}) \times \{R, L, S\}$: De transitierelatie. Deze bepaalt wat er kan gebeuren vanuit een gegeven toestand $\in Q$, wanneer een symbool $\in \Gamma$ wordt gelezen.

(Uit FI3, college 3)

Cook-Levin: SAT is NP-Volledig

De formule ϕ heeft een hoop Boolse variabelen:

- Q_t^q: Op tijdstip t is de machine in toestand q
  (voor $0 \leq t \leq N(|x|)$ en elke toestand $q \in Q$)
- $H_{t,j}$: Op tijdstip t scant de machine cel j
  (voor $0 \leq t \leq N(|x|)$ en $-N(|x|) \leq j \leq N(|x|)$)
- $S_{t,j}^a$: Op tijdstip t bevat de cel j symbool a
  (voor $0 \leq t \leq N(|x|)$, $-N(|x|) \leq j \leq N(|x|)$ en $a \in \Gamma \cup \{\Delta\}$)

Cook-Levin: SAT is NP-Volledig

De formule ϕ is een conjunctie van de volgende eisen:

1. De TM start in tijdstip $t = 0$ in de initiële configuratie
2. Elke configuratie is "geldig":
   - 2.1 De TM is in één toestand
   - 2.2 De TM scant één cel
   - 2.3 Elke cel bevat één symbool
3. Elke stap verloopt volgens de transitiefunctie δ van de TM
4. De TM eindigt in de accepterende toestand

Cook-Levin: SAT is NP-Volledig

De formule ϕ is een conjunctie van de volgende clausules:

1.1 $Q_0^{q_0}$
De TM start (op tijdstip $t = 0$) in toestand q_0

1.2 $H_{0,1}$
De machine start (op tijdstip $t = 0$) in toestand q_0 op tape positie 1

1.3 $S_{0,j}^{x_j}$ voor $1 \leq j \leq |x|$
Tape bevat x_j op positie j op tijdstip $t = 0$.

1.4 $S_{0,j}^{\Delta}$ voor $j \in \{-N, \ldots, 0, |x|+1, \ldots, N\}$
Tape bevat blanco symbool Δ op de andere posities op tijdstip $t = 0$

2.1 $(Q_t^{q_a} + Q_t^{q_r} + Q_t^{q_0} + Q_t^{q_1} + \cdots + Q_t^{q_m} = 1)$ voor $-N \leq t \leq N$
Ten alle tijden is de TM in één toestand

2.2 $(H_{t,-N} + \cdots + H_{t,N-1} + H_{t,N} = 1)$ voor $-N \leq t \leq N$
Ten alle tijden scant de TM precies één tape cel

2.3 $(S_{t,j}^{a_1} + S_{t,j}^{a_2} + \cdots + S_{t,j}^{a_{|\Gamma|}} = 1)$ voor $-N \leq t \leq N$
Ten alle tijde bevat elke cel precies één symbool

3.1 $Q_t^p \wedge H_{t,j} \wedge S_{t,j}^a \implies \bigvee_{(p,a,q,b,d)\in\delta} \left(Q_{t+1}^q \wedge H_{t+1,j+d} \wedge S_{t+1,j}^b \right)$
Elke stap verloopt volgens de transitie-relatie δ

3.2 $(S_{t,j}^a \wedge \neg H_{t,j} \implies S_{t+1,j}^a)$
Een cel die op tijdstip t niet gescand wordt, bevat op tijdstip $t+1$ hetzelfde symbool

---

4 $Q_N^{q_a}$
Op tijdstip N stopt de berekening in de accepterende toestand

Stelling van Savitch

Zij $s(n)\colon \mathbb{Z}_{\geq 0} \to \mathbb{Z}_{\geq 0}$ een functie.

Definition (Space)

$\mathrm{SPACE}(s(n)) = \{L \subseteq \{0,1\}^* |$ Er is een deterministische Turing Machine die L beslist en hooguit $s(|x|)$ tape symbolen gebruikt$\}$.

$\mathrm{NSPACE}(s(n)) = \{L \subseteq \{0,1\}^* |$ Er is een non-deterministische Turing Machine die L beslist en, op elk berekeningspad, hooguit $s(|x|)$ tape symbolen gebruikt$\}$.

Stelling van Savitch: $\mathrm{NSPACE}(s(n)) \subseteq \mathrm{SPACE}(s(n)^2)$

Vergelijk met: $\mathrm{TIME}(t(n)) \subseteq \mathrm{NTIME}(2^{t(n)})$

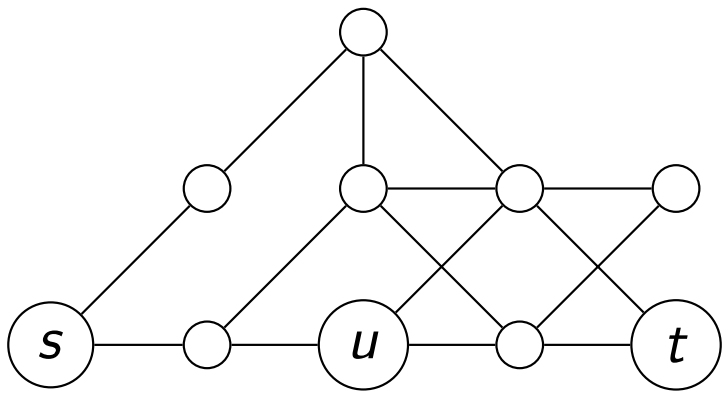
Stelling van Savitch: Configuratiegraaf

Zeg dat een tupel (Tape, q, pos) een *configuratie* is van een TM. Er is een initiële configuratie $(x, q_0, 1)$ en een eindconfiguratie, $(\Delta, q_a, 1)$.
De **configuratiegraaf** is een graaf met takken (s, t) z.d.d. $s \vdash_T t$, i.e., T kan in één stap van configuratie s naar configuratie t.

Stelling van Savitch: Bereikbaarheid

Laat $d(s,t)$ de afstand van knoop s naar t zijn, en $\ell \geq 1$ een getal. Dan geldt:

$$d(s,t) \leq \ell \iff (s,t) \in E \text{ of } \exists u \in V : d(s,u) \leq \frac{1}{2}\ell \wedge d(u,t) \leq \frac{1}{2}\ell$$



Algoritme van Savitch

```
1:  procedure SAVITCH(G, s, t, ℓ)
2:      if ℓ = 1 then
3:          return (s, t) ∈ E_G
4:      else
5:          for u ∈ V_G do
6:              if SAVITCH(G, s, u, ½ℓ) en SAVITCH(G, u, t, ½ℓ) then
7:                  return TRUE
8:              end if
9:          end for
10:     end if
11:     return FALSE
12: end procedure
```

Output: $\textsc{Savitch}(G, s, t, \ell) = \textsc{True}$ desda er een pad van s naar t is van lengte hooguit ℓ.
Roep aan als: $\textsc{Savitch}(G, s, t, |V_G|)$.

Algoritme van Savitch

```
1:  procedure SAVITCH(G, s, t, ℓ)
2:      if ℓ = 1 then
3:          return (s, t) ∈ E_G
4:      else
5:          for u ∈ V_G do
6:              if SAVITCH(G, s, u, ½ℓ) en SAVITCH(G, u, t, ½ℓ) then
7:                  return TRUE
8:              end if
9:          end for
10:     end if
11:     return FALSE
12: end procedure
```

Geheugengebruik:

$$S(\ell) = \begin{cases} \mathcal{O}(1) & \text{Als } \ell = 1 \\ \log_2(|V_G|) + S(\frac{1}{2}\ell) & \text{Als } \ell \geq 2 \end{cases} = \mathcal{O}(\log_2(|V_G|)^2) \quad (1)$$

Algoritme van Savitch

Algoritme	Geheugen	Tijd
Savitch	$\mathcal{O}(\log(n)^2)$	$\mathcal{O}(2^{\log(n)^2}) = \mathcal{O}(n^{\log(n)})$
Dijkstra	$\mathcal{O}(n \log n)$	$\mathcal{O}(n)$

Algoritme van Savitch

Theorem (Stelling van Savitch)

$\mathrm{NSPACE}(s(n)) \subseteq \mathrm{SPACE}\left(s(n)^2\right)$.

Bewijs.

Zij $L \in \mathrm{NSPACE}(s(n))$ een taal en $x \in \{0,1\}^*$ een string. Laat $n = |x|$ Neem een Turing Machine T die L beslist in $s(n)$ ruimte.

Dan is $x \in L$ desda er een pad is vanaf de initiële configuratie naar een accepterende configuratie. T komt in hooguit $|\Gamma_T|^{s(n)} \times |Q_T|$ verschillende configuratie. Dus het pad heeft een lengte van hooguit $|\Gamma_T|^{s(n)} \times |Q_T| \times s(n)$. Het algoritme van Savitch beslist dat met $\mathcal{O}(\log\left(|\Gamma_T|^{s(n)} \times |Q_T| \times s(n)\right)^2) = \mathcal{O}(s(n)^2)$ geheugen. □

Algoritme van Savitch: Gevolgen

Theorem (Stelling van Savitch)
$\mathrm{NSPACE}(s(n)) \subseteq \mathrm{SPACE}\left(s(n)^2\right)$.

Definitie: $\mathrm{PSPACE} = \bigcup_{c=1}^{\infty} \mathrm{SPACE}(n^c)$

Gevolg. $\mathrm{PSPACE} = \mathrm{NPSPACE}$

Gevolg. $\mathrm{EXPSPACE} = \mathrm{NEXPSPACE}$

Gevolg. Quantified Boolean Formula is een PSPACE-Volledig probleem.

Algoritme van Savitch: Quantified Boolean Formulas

Een *gekwantificeerde* Boolse formule kennen we uit Logica:

Voorbeeld:

$$\phi = \underbrace{\forall x : \exists y : \forall z}_{\text{Quantifiers}} : (x \vee z) \wedge (y \oplus z) \tag{2}$$

Algoritme van Savitch: Quantified Boolean Formulas

Een gekwantificeerde Boolse formule is *waar* of *onwaar*.

$$\forall x \in \{0,1\} : \exists y \in \{0,1\} : (x \vee \neg y) \wedge (\neg x \vee y) \quad \textbf{(Waar)} \quad (3)$$

$$\exists x \in \{0,1\} : \forall y \in \{0,1\} : (x \vee y) \wedge (\neg x \vee \neg y) \quad \textbf{(Onwaar)} \quad (4)$$

Input: Een gekwantificeerde Boolse formule
Output: Is de formule waar?

Algoritme van Savitch: Quantified Boolean Formulas

Gegeven een Turing Machine T kunnen we een formule $\psi_0(s,t)$ maken met als input (encodering van) configuraties s,t zdd $\psi_0(s,t)$ waar is desda T vanuit s naar t kan:

$$\psi_0(s,t) \iff s \vdash_T t \tag{5}$$

Maak een formule ψ_ℓ die waar is desda $s \vdash_T^k t$ voor een $k \leq 2^\ell$. Deze formule heeft alleen de $\exists$ quantifier:

$$\psi_\ell(s,t) = \exists u\colon \psi_{\ell-1}(s,u) \wedge \psi_{\ell-1}(u,t) \tag{6}$$

Deze formule is (helaas) $\Theta(2^\ell)$ symbolen lang. Repareer met extra quantifiers:

$$\psi_\ell(s,t) = \exists u\colon \forall d,e\colon\ (s=d \wedge u=e \vee u=d \wedge t=e) \implies \psi_{\ell-1}(d,e) \tag{7}$$

Algoritme van Savitch: Quantified Boolean Formulas

Gevolg. Waarheid van een gekwantificeerde Boolse formule is PSPACE-Hard.
Nu: $\mathrm{QBF} \in \mathrm{PSPACE}$ want er is een algoritme met $\mathcal{O}(n^2)$ ruimte.

Quantified Boolean Formulas: Algoritme

1: **procedure** $\mathrm{QBF}(\psi = Q_1 x_1 Q_2 x_2 \cdots Q_n x_n : \phi(x_1, \ldots, x_n))$
2: **if** $n = 0$ **then**
3: **return** $\phi = \textsc{True}$
4: **else**
5: $\psi_0 := Q_2 x_2 \cdots Q_n x_n : \phi(0, x_2, \ldots, x_n)$ ▷ Omschrijven
6: $\psi_1 := Q_2 x_2 \cdots Q_n x_n : \phi(1, x_2, \ldots, x_n)$ ▷ Omschrijven
7: $a := QBF(\psi_0)$ ▷ Recursie
8: $b := QBF(\psi_1)$ ▷ Recursie
9: **if** $Q_1 = \exists$ **then**
10: **return** $a \vee b$
11: **else**
12: **return** $a \wedge b$
13: **end if**
14: **end if**
15: **end procedure**

Tijd: $\mathcal{O}(n2^n)$
Ruimte: $\mathcal{O}(n)$

Algoritme van Savitch: Gevolgen

Theorem (Stelling van Savitch)
$\mathrm{NSPACE}(s(n)) \subseteq \mathrm{SPACE}\left(s(n)^2\right)$.

Definitie: $\mathrm{PSPACE} = \bigcup_{c=1}^{\infty} \mathrm{SPACE}(n^c)$

Gevolg. $\mathrm{PSPACE} = \mathrm{NPSPACE}$

Gevolg. $\mathrm{EXPSPACE} = \mathrm{NEXPSPACE}$

Gevolg. Quantified Boolean Formula is een PSPACE-Volledig probleem.

Klassificatie van talen tot nu toe

Taal: Een verzameling $L \subseteq \{0,1\}^*$.

$$\underbrace{\text{REGULAR} \subsetneq \text{CFL}}_{\text{FI 1+2}} \subsetneq \text{P} \subseteq \text{NP} \subseteq \text{PSPACE} = \text{NPSPACE} \subseteq \text{EXP} \subsetneq \underbrace{\text{R} \subsetneq \text{RE}}_{\text{FI 3}}$$

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)