NL-Vollständigkeit

- R-many-one-Reduktion

Sei A, B beliebige Problem

Es gilt $A \leq_m^R B$ gdw. $\exists f \in R$:

$w \in A \iff f(w) \in B$.

- Ein Problem B ist C-hart (C-Schwer) bzgl. R-many-one-Reduction gdw.

$\forall A \in C: A \leq_m^R B$

B ist C-vollständig, wenn es in C liegt <u>und</u> C-Schwer ist.

Was ist NL?

Klasse der Probleme, die nicht deterministisch in logspace gelöst werden können.

<u>Beh:</u> PATH ist NL-Vollständig.

<u>Eingabe:</u> Gerichteter Graph G, Knoten s und t.

<u>Frage:</u> Gibt es einen Pfad von s nach t.

① <u>ACCESSIBLE (Acc):</u>

<u>Eingabe:</u> Grammatik $G = (N, \Sigma, P, S)$, Nichtterminal B.

<u>Frage:</u> $S \Rightarrow^* \alpha B \beta$ $(\alpha, \beta \in (N \cup \Sigma)^*)$

① <u>$ACC \leq_m^{log} PATH$</u>

Sei $G = (N, \Sigma, P, S)$ und ein Nichtterminal B gegeben.

<u>Idee:</u> Baue Graphen aus Nichtterminalen N und verbinde zwei NT C und D, wenn es eine Reduktion $C \rightarrow \alpha D \beta$ gibt.

Reduktion: $f(G,B) = (H,S,B)$ mit

(S: Start knoten, B: End knoten)

$H = (V,E)$ wobei

$V = N$ und $(C,D) \in E$ gdw. es gibt eine Produktion $C \to \alpha D \beta$ gibt.

f ist logspace berechenbar.

Offensichtlich gilt: $(G,B) \in ACC \Leftrightarrow f(G,B) \in PATH$.

$\Rightarrow$ Es gilt $ACC \in NL$.

Geht das in Logspace? - Ja, Warum?

Die Konten V werden von N kopiert.

COPY passiert in Logspace.

② $PATH \leq_m^{log} ACC$.

Sei $H = (V,E)$ ein gerichteter Graph, s und t Knoten.

Reduktion: $f(H,s,t) = (G,B) = ((N, \Sigma, P, S), B)$ mit

$N = V$

$\Sigma = \{0\}$

$P = E$

$S = s$

$B = t$

$\Rightarrow$ ACC ist NL-Schwer, also NL-Vollständig.

② s-t-Cycle

Eingabe: ein gerichteter Graph und Knoten $s, t \in V$.

- Frage: Gibt es Kreis, der s und t enthält?

s-t-Cycle $\in$ NL:

Löse PATH(G, s, t) und ~~PATH~~ PATH(G, t, s).

Gibt es die beiden Pfade? Dann gibt es einen Kreis.

1) s-t-Cycle ist NL-Schwer

$(PATH \leq_m^{log} s\text{-}t\text{-}CYCLE)$

Sei (G, s, t) gegeben

Reduktion: $f(G, s, t) = (G', s, t)$, wobei $G' = (V, E \cup (t, s))$.

Eine Kante einfügen $\Rightarrow$ Logspace berechenbar.

Korrektheit:

$(G, s, t) \in PATH \Rightarrow (G', s, t) \in PATH$ und $(G', t, s) \in PATH$

$\Rightarrow (G', s, t) \in s\text{-}t\text{-}CYCLE.$

$(G, s, t) \notin PATH \Rightarrow (G', s, t) \notin PATH.$

$\Rightarrow (G', s, t) \notin s\text{-}t\text{-}CYCLE$

$\Rightarrow$ s-t-CYCLE NL-Vollständig

Wieso offensichtlich NL-machbar?

Erstmal Reduktion geht hier nicht. Reduktion in PATH muss genau einmal verwendet werden.

③ REG - EMPTY

Eingabe: Ein Automat (DFA / NFA)

$A = (Q, \Sigma, \rightarrow, q_0, Q_F)$

Frage: Ist $L(A) = \emptyset$?

REG-EMPTY $\in$ NL

Statt direkt zu zeigen es NL, zeige $\overline{\text{REG-EMPTY}} \in$ NL ist.

I&S Satz NL = coNL

$\overline{\text{REG-EMPTY}}$ = REG-NONEMPTY $\in$ NL

Idee: $L(A) \neq \emptyset$ gdw. es einen Pfad von q_0 zu einem $q_f \in Q_F$ gibt.

Algo:

1) Rate: $q_f \in Q_F$.
2) Löse PATH(G_A, q_0, q_f)

offensichtlich in logspace.

Nun zeigen wir REG-NONEMPTY ist NL-Schwer. (PATH $\leq_m^{\log}$ REG-NONEMPTY)

Reduktion $f(G, s, t) = A = (Q, \Sigma, \rightarrow, q_0, Q_F)$

mit
$Q = V$
$\Sigma = \{0\}$
$x \xrightarrow{0}_A v$ gdw. $(x, v) \in E$
$q_0 = s$
$Q_F = \{t\}$

Wann ist ein Sprache L leer?

⇒ Wenn da kein Endzustand gibt.

⇒ REG-NONEMPTY ist NL-Vollständig

$\overset{NL = coNL}{\Rightarrow}$ REG-EMPTY ist NL-Vollständig