Übungsstunde 10

Samstag, 25. November 2023 14:49

Nachbesprechung Serie 8

EE-Reduktionen: 1) "Wir simulieren M auf w" gibt **0 Punkte** an der Klausur

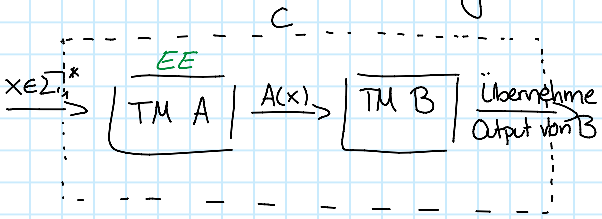
Ihr müsst schreiben: 2) "Wir generieren die Kodierung einer TM M' welche unabhängig von ihrer Eingabe M auf w simuliert"

Bei Variante 1) ist nicht garantiert, dass die TM immer hält (Bedingung an eine EE-Reduktion

Was macht eine EE-Reduktion Schematisch

Die TM aus der EE-Konstruktion welche immer hält nennen wir A.

Wir wollen $L_1 \leq_{EE} L_2$ zeigen.



$x \in L_1 = L(C) \iff A(x) \in L_2 = L(B)$

TM A: Das ist die EE-Red. aka die Transformation der Eingabe

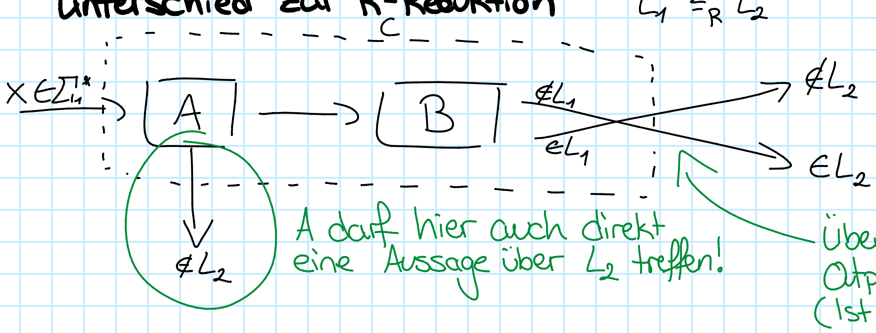
TM B: Die TM B prüft einfach ob $A(x) \in L_2$ und **übernimmt** diese Ausgabe

TM C: Die TM welche zuerst A und dann B ausführt

Diese vernachlässigen wir bei EE-Reduktionen.

Nur wenn wir aus der EE-Red. $\in \mathcal{L}_R$ oder $\notin \mathcal{L}_R$ folgern wollen müssen wir diese wie bei R-Red. angeben.

Unterschied zur R-Reduktion $L_1 \leq_R L_2$



1. $L(C) = L_2$ unter $L(B) = L_1$ und B Algo!
2. C hält immer

A darf hier auch direkt eine Aussage über L_2 treffen!

Übernehmen den Output nicht! (Ist bei EE-nicht erlaubt)

$\ulcorner L_1 \leq_R L_2 \iff \ulcorner L_2 \in \mathcal{L}_R \implies L_1 \in \mathcal{L}_R \urcorner$

$\iff \neg(L_2 \in \mathcal{L}_R) \vee (L_1 \in \mathcal{L}_R)$ $\Leftarrow$ Falls $L_1 \notin \mathcal{L}_R \implies L_2 \notin \mathcal{L}_R$

Auch wenn dieser Ansatz logisch i.O. ist gibt er an der Prüfung 0 Pkt $\lrcorner$

Aufgabe 24 b): Finde fehler

Angenommen $w = Kod(M)\#x$.

Dann $A(x) = Kod(M')$, wobei M' unabhängig von ihrer Eingabe M auf x simuliert.

Falls M hält $\Rightarrow$ M' verwirft

Falls M nicht hält $\Rightarrow$ M' akzeptiert ↯

M' kann nicht nachvollziehen ob M auf x hält oder nicht – dies ginge nur in einer R-Red. wo wir annehmen, dass M' ein Algo. ist welcher entscheidet ob eine TM hält oder nicht $\Rightarrow$ $L(M') = \emptyset$ hier, da M hält nicht $\Rightarrow$ M' hält nicht

Variante 2: A konstruiert eine TM M', welche für jede Eingabe M auf x simuliert für $|y|$ Schritte, wobei y nicht-det. gewählt wird.

Hält M innerhalb der $|y|$-Schritte $\Rightarrow$ verwirf ↯

Sonst: akzeptier

Das Problem hier ist, dass wir bei NEA's immer nur akzeptierende Bedingungen raten können (i.e. sowas wie wenn nach |y|-Schritten term. $\Rightarrow$ akzeptiere). Da wir nur Wörter akz. die nicht halten, existiert keine obere Grenze an Berechnungs-schritten, nach denen wir das gelesene Wort akzeptieren können, aka $\nexists$ Lösung welche wir nicht-det. raten könnten.

Richtig: Für seine Eingabe y simuliert M' die TM M auf x für $|y|$ Schritten. Terminiert diese $\Rightarrow$ M' verwirft, sonst akzeptiere.

Wieso geht das?

z: $x \in L_H^c \Leftarrow f(x) \in L_{all}$

Per Kontraposition. $x \notin L_H^c \Rightarrow f(x) \notin L_{all}$

$\Longleftrightarrow x \in L_H \Rightarrow x \notin L_{all}$

$x \in L_H \Rightarrow \exists k \in \mathbb{N}$ s.d. M auf x in k Schritten terminiert.

$\Rightarrow \forall w \in \Sigma_2^{\geq k}$ verwirft M' $\Rightarrow Kod(M') \notin L_{all}$

Da L_{all} alle Wörter akzeptiert!

$NTIME(f) \stackrel{\smile}{=} \{ L(M) \mid M \text{ NMTM mit } Time_M(n) \in \mathcal{O}(f(n)) \}$

<u>Beh.</u>: a) NTIME(f) ist abgeschlossen unter Vereinigung $(L_1, L_2 \in NTIME(f) \Rightarrow L_1 \cup L_2 \in NTIME(f))$

<u>Beweis</u>: NMTM muss halten (Wenn nicht sicher ob 2 NMTM halten → Parallel laufen lassen)

M_1 NMTM s.d. $L(M_1) = L_1$ mit k_1-Bändern & $Time_{M_1}(w_1) \in \mathcal{O}(f(n))$, $|w_1| = n$

M_2 NMTM s.d. $L(M_2) = M_2$ mit k_2-Bändern & $Time_{M_2}(w_2) \in \mathcal{O}(f(n))$, $|w_2| = n$

Konstruiere nun eine NMTM M auf $(k_1 + k_2 + 1)$-Bändern für $L = L_1 \cup L_2$

M arbeitet wie folgt: M simuliert M_1 auf Band 2 bis k_1+1 (1. Band ist Eingabeband)

(Parallele simulation) M simuliert M_2, auf Band k_1+2 bis k_1+k_2+1

M akzeptiert $\Leftrightarrow$ M_1 oder M_2 (oder beide) akzeptiert

$\exists$ akzeptierende Berechnung auf M gdw. $\exists$ (akz. Berechnung in M_1 oder M_2)

Zeitkomplexität: M kopiert Input auf jeweils das erste Band von M_1 & M_2. Danach stellt M die Köpfe von M_1 & M_2 auf den Anfang.

Da $|Input| = n$ ist dies in $\mathcal{O}(2n) = \mathcal{O}(n)$ möglich.

Eine akzeptierende Berechnung von M benötigt nicht mehr Zeit als "Zeit zum Kopieren der Eingabe" + "Zeit der kürzesten Berechnung von M_1 oder M_2"

$Time_M(w) \in \mathcal{O}(n) + \min\{ Time_{M_1}(w), Time_{M_2}(w) \} \in \mathcal{O}(f(n))$

$\Rightarrow L = L_1 \cup L_2 \in NTIME(f(n))$. $\square$