Theoretische Informatik HS23

Nicolas Wehrli

Übungsstunde 09

25. November 2023

ETH Zürich
nwehrl@ethz.ch

Heute

1. Feedback zur Serie
2. Satz von Rice - Beweis
3. EE Reduktion angewendet für $\mathcal{L}_{\mathrm{RE}}$
4. Reduktionsaufgaben
5. Komplexitätstheorie Einführung

Feedback zur Serie

Feedback zur Serie

- Recht gut.
- Wenn in einer Reduktion eine Algorithmus A für ein beliebiges L' annehmt, dann dürft ihr **nichts** über A annehmen.
    i. Ihr dürft nicht in A hineingreifen und irgendetwas ändern.
    ii. Ihr müsst die Wörter von der richtigen Form übergeben (i.e. genau so wie sie in der Sprache beschrieben sind).
    iii. Beispiel:
    Sei A so dass $L(A) = L_{\mathrm{H}}$, dann übergebt ihr die Wörter $\mathrm{Kod}(M)$ und w nicht separat, sondern ihr übergebt genau ein Wort nämlich $\mathrm{Kod}(M)\#w$.

Satz von Rice - Beweis

Prerequisites

Zur Erinnerung:

Semantisch nichttriviales Entscheidungsproblem über TMs

Das Entscheidungsproblem (Σ, L), bzw. die Sprache L muss folgendes erfüllen.

I. $L \subseteq \textbf{KodTM}$
II. $\exists M_1$ so dass $\text{Kod}(M_1) \in L$(i.e. $L \neq \emptyset$)
III. $\exists M_2$ so dass $\text{Kod}(M_2) \notin L$(i.e. $L \neq \textbf{KodTM}$)
IV. Für zwei TM A und B mit $L(A) = L(B)$ gilt

$$\text{Kod}(A) \in L \iff \text{Kod}(B) \in L$$

KodTM $\subseteq (\Sigma_{\text{bool}})^*$ ist die Menge aller Kodierungen von Turingmaschinen.

Wir brauchen

Lemma 5.8

$$L_{H,\lambda} \notin \mathcal{L}_{\mathrm{R}}$$

Zur Erinnerung:

$$L_{H,\lambda} = \{\mathrm{Kod}(M) \mid M \text{ hält auf } \lambda\}$$

Idee

Wir zeigen für jedes **semantisch nichtriviale Entscheidungsproblem** (Σ, L)

$$L \in \mathcal{L}_\mathrm{R} \implies L_{H,\lambda} \in \mathcal{L}_\mathrm{R}$$

Aus dem folgt dann per Kontraposition

$$L_{H,\lambda} \notin \mathcal{L}_\mathrm{R} \implies L \notin \mathcal{L}_\mathrm{R}$$

Mit der Aussage $L_{H,\lambda} \notin \mathcal{L}_\mathrm{R}$ von **Lemma 5.8**, können wir dann

$$L \notin \mathcal{L}_\mathrm{R}$$

wie gewünscht folgern.

Idee

Wir müssen noch die Implikation

$$L \in \mathcal{L}_{\mathrm{R}} \implies L_{H,\lambda} \in \mathcal{L}_{\mathrm{R}}$$

beweisen.

Kernidee

Wir zeigen die **Existenz** einer Reduktion, aus der die Implikation folgt.

Idee

Konkret machen wir eine **Case Distinction** und zeigen jeweils

- Die **Existenz** einer EE-Reduktion von $L_{H,\lambda}$ auf L
  Daraus folgt $L_{H,\lambda} \leq_{\text{EE}} L$.
- oder die **Existenz** einer EE-Reduktion $L_{H,\lambda}$ auf $L^{\complement}$
  Daraus folgt $L_{H,\lambda} \leq_{\text{EE}} L^{\complement}$.

Zur Erinnerung:

Lemma 5.3

Seien $L_1 \subseteq \Sigma_1^*$ und $L_2 \subseteq \Sigma_2^*$ zwei Sprachen.

$$L_1 \leq_{\text{EE}} L_2 \implies L_1 \leq_{\text{R}} L_2$$

Idee

Weshalb reicht es $L_{H,\lambda} \leq_{\text{EE}} L^{\complement}$ zu zeigen?

Lemma 5.4

Sei Σ ein Alphabet. Für jede Sprache $L \subseteq \Sigma^*$ gilt:

$$L \leq_{\text{R}} L^{\complement} \text{ und } L^{\complement} \leq_{\text{R}} L$$

In beiden Cases folgt mit **Lemma 5.3** und **Lemma 5.4**, die gewünschte Aussage $L_{H,\lambda} \leq_{\text{R}} L$.

Idee

Explizit gilt nun

1. $$L_{H,\lambda} \leq_{\mathrm{EE}} L^{\complement} \xRightarrow{\textbf{Lemma 5.3}} L_{H,\lambda} \leq_{\mathrm{R}} L^{\complement} \xRightarrow{\textbf{Lemma 5.4}} L_{H,\lambda} \leq_{\mathrm{R}} L$$

2. $$L_{H,\lambda} \leq_{\mathrm{EE}} L \xRightarrow{\textbf{Lemma 5.3}} L_{H,\lambda} \leq_{\mathrm{R}} L$$

Aus $L_{H,\lambda} \leq_{\mathrm{R}} L$ folgt (in beiden Cases) die gewünschte Implikation

$$L \in \mathcal{L}_{\mathrm{R}} \implies L_{H,\lambda} \in \mathcal{L}_{\mathrm{R}}$$

Beweis

Sei $M_\emptyset$ eine TM s.d. $L(M_\emptyset) = \emptyset$.

Case Distinction

I. $\mathbf{Kod}(\mathbf{M}_\emptyset) \in \mathbf{L}$
   Wir zeigen $L_{H,\lambda} \leq_{\mathrm{EE}} L^{\complement}$.
II. $\mathbf{Kod}(\mathbf{M}_\emptyset) \notin \mathbf{L}$
   Wir zeigen $L_{H,\lambda} \leq_{\mathrm{EE}} L$.

Case I. $\mathbf{Kod}(\mathbf{M}_\emptyset) \in \mathbf{L}$

Es **existiert** eine TM $\overline{M}$, so dass $\text{Kod}(\overline{M}) \notin L$. (Nichttrivialität)

Wir beschreiben eine TM S, so dass für eine Eingabe $x \in (\Sigma_{\text{bool}})^*$

$$x \in L_{H,\lambda} \iff S(x) \in L^{\complement}$$

Daraus folgt dann die gewünschte EE-Reduktion.

Wir verwenden dabei $M_\emptyset$ und $\overline{M}$, da $\text{Kod}(M_\emptyset) \notin L^{\complement}$ und $\text{Kod}(\overline{M}) \in L^{\complement}$.

Case I. Kod$(\mathbf{M}_\emptyset) \in$ L - Beschreibung von S

Eingabe $x \in (\Sigma_{\text{bool}})^*$

1. S überprüft ob $x = \text{Kod}(M)$ für eine TM M.
   Falls dies **nicht** der Fall ist, gilt $S(x) = \text{Kod}(M_\emptyset)$
2. Sonst $x = \text{Kod}(M)$. Dann $S(x) = \text{Kod}(A)$, wobei A wie folgt kodiert ist.
   i. Gleiches Eingabealphabet wie $\overline{M}$, i.e. $\Sigma_A = \Sigma_{\overline{M}}$.
   ii. Für eine beliebige Eingabe $y \in (\Sigma_{\overline{M}})^*$, simuliert A zuerst M auf λ **ohne die Eingabe** y **zu überschreiben**.
   iii. Danach simuliert A die TM $\overline{M}$ auf die gegebene Eingabe y.
   iv. Akzeptiert y genau dann, wenn $\overline{M}$ y akzeptiert.

Korrektheit

Wir zeigen

$$x \in L_{H,\lambda} \iff S(x) \in L^{\complement}$$

$(\implies)$:

Wir nehmen $x \in L_{H,\lambda}$ an und zeigen $S(x) \in L^{\complement}$.

Da M auf λ hält, wird A immer $\overline{M}$ auf der Eingabe y simulieren und wir haben $L(A) = L(\overline{M})$.

Da L (und somit auch $L^{\complement}$) ein **semantisches** Entscheidungsproblem ist, gilt

$$\mathrm{Kod}(\overline{M}) \in L^{\complement} \implies \mathrm{Kod}(A) \in L^{\complement}$$

Da die LHS der Implikation gegeben ist, folgt $S(x) = \mathrm{Kod}(A) \in L^{\complement}$

Korrektheit

$(\Longleftarrow)$:

Wir nehmen $x \notin L_{H,\lambda}$ an und zeigen $S(x) \notin L^{\complement}$.

Aus Kontraposition folgt dann die gewünschte Rückimplikation.

Da M nicht auf λ hält, wird A bei jeder Eingabe nicht halten.

Somit folgt $L(A) = L(M_{\emptyset})$ und da $\mathrm{Kod}(M_{\emptyset}) \notin L^{\complement}$ per semantische Eigenschaft von L

$$S(x) = \mathrm{Kod}(A) \notin L^{\complement}$$

Case II.

Zweite Case funktioniert genau gleich.

Wir haben $\mathrm{Kod}(M_\emptyset) \notin L$.

Per Nichttrivialität existiert eine TM $\overline{M}$ mit $\mathrm{Kod}(\overline{M}) \in L$.

...

$\blacksquare$

EE Reduktion angewendet für $\mathcal{L}_{\mathrm{RE}}$

Lemma zu RE-Reduktion

EE-Reduktion impliziert RE-Reduktion (**nicht in der Vorlesung**)

$$L_1 \leq_{\text{EE}} L_2 \implies (L_2 \in \mathcal{L}_{\text{RE}} \implies L_1 \in \mathcal{L}_{\text{RE}})$$

Beweis

Sei $L_1 \leq_{\text{EE}} L_2$ und $L_2 \in \mathcal{L}_{\text{RE}}$.

Wir zeigen nun $L_1 \in \mathcal{L}_{\text{RE}}$.

Per Definition von $L_1 \leq_{\text{EE}} L_2$ existiert ein Algorithmus F, der die Funktion $f : \Sigma_1^* \to \Sigma_2^*$ berechnet, so dass

$$\forall x \in \Sigma_1^*. x \in L_1 \iff f(x) \in L_2$$

Lemma zu RE-Reduktion

Da $L_2 \in \mathcal{L}_{\text{RE}}$ existiert eine TM M_2 (die nicht unbedingt immer terminiert) mit $L(M_2) = L_2$.

Wir beschreiben mit F und M_2 nun eine TM M_1 mit $L(M_1) = L_1$.

Eingabe: $x \in \Sigma_1^*$

1. F berechnet auf x und übergibt seine Ausgabe $f(x)$ zur TM M_2
2. M_2 berechnet auf $f(x)$ und die Ausgabe wird übernommen.

Lemma zu RE-Reduktion

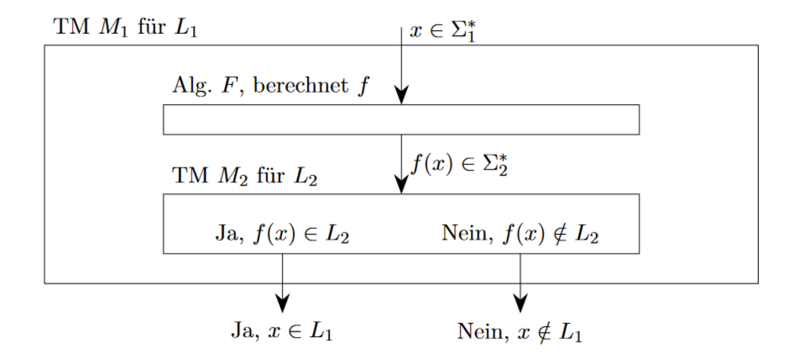

Abbildung 1: TM M_1, Zsf. Fabian Frei

Korrektheit ($L_1 = L(M_1)$)

Case Distinction

I. $\mathbf{x} \in \mathbf{L_1}$

$\implies f(x) \in L_2$ (Algorithmus F terminiert immer)

$L(M_2) = L_2 \implies f(x) \in L(M_2)$

da die Ausgabe von M_2 übernommen wird

$\implies x \in L(M_1)$

II. $\mathbf{x} \notin \mathbf{L_1}$

$\implies f(x) \notin L_2$

$\implies f(x) \notin L(M_2)$

$\implies x \notin L(M_1)$

$\blacksquare$

Reduktionsaufgaben

Aufgabe 1

Zeige

$$L_{\text{diag}} \leq_{\text{EE}} L_{\text{H}}^{\complement}$$

Zur Erinnerung:

$$L_{\text{diag}} = \{w_i \in (\Sigma_{\text{bool}})^* \mid M_i \text{ akzeptiert } w_i \text{ nicht}\}$$

$$\begin{aligned} L_{\text{H}}^{\complement} = {} & \{\text{Kod}(M)\#w \in \{0,1,\#\}^* \mid M \text{ hält nicht auf } w\} \\ & \cup \{x \in \{0,1,\#\}^* \mid x \text{ nicht von der Form Kod}(M)\#w\} \end{aligned}$$

Lösung 1

Wir beschreiben einen Algorithmus A, so dass

$$x \in L_{\text{diag}} \iff A(x) \in L_{\text{H}}^{\complement}$$

Eingabe: $x \in (\Sigma_{\text{bool}})^*$

1. Findet i so dass $x = w_i$
2. Generiert $\text{Kod}(M_i)$
3. Generiert $\text{Kod}(\overline{M}_i)$ mit folgenden Modifikationen zu $\text{Kod}(M_i)$
   - Transitionen nach q_{reject} werden in eine Endlosschleife umgeleitet.
4. Gibt $\text{Kod}(\overline{M}_i)\#w_i$ aus.

Lösung 1

Case Distinction

I. $\mathbf{x} \in \mathbf{L}_{\text{diag}}$

$$\begin{aligned} &\Longrightarrow M_i \text{ akzeptiert } x = w_i \text{ nicht} \\ &\Longrightarrow \overline{M}_i \text{ hält nicht auf } w_i \\ &\Longrightarrow A(x) = \text{Kod}(\overline{M}_i)\#w_i \in L_{\text{H}}^{\complement} \end{aligned}$$

II. $\mathbf{x} \notin \mathbf{L}_{\text{diag}}$

$$\begin{aligned} &\Longrightarrow M_i \text{ akzeptiert } x = w_i \\ &\Longrightarrow \overline{M}_i \text{ hält auf } w_i \\ &\Longrightarrow A(x) = \text{Kod}(\overline{M}_i)\#w_i \notin L_{\text{H}}^{\complement} \end{aligned}$$

$\blacksquare$

Aufgabe 2

Zeige

$$L_{\mathrm{U}}^{\complement} \leq_{\mathrm{EE}} L_{\mathrm{diag}}$$

Komplexitätstheorie Einführung

Time

Sei M eine MTM oder TM, die immer hält. Sei Σ das Eingabealphabet von M. Sei $x \in \Sigma^*$ und $D = C_1, C_2, ..., C_k$ die Berechnung von M auf x.
Die **Zeitkomplexität $\mathbf{Time_M(x)}$ der Berechnung von M auf x** ist definiert durch

$$\mathbf{Time_M(x)} = k - 1.$$

Die **Zeitkomplexität von M** ist die Funktion $\text{Time}_M : \mathbb{N} \to \mathbb{N}$, definiert durch

$$\mathbf{Time_M(n)} = \max \left\{ \text{Time}_M(x) \mid x \in \Sigma^n \right\}.$$

Space

Sei $k \in \mathbb{N} \setminus \{0\}$. Sei M eine k-Band-TM, die immer hält.
Sei

$$C = (q, x, i, \alpha_1, i_1, \alpha_2, i_2, ..., \alpha_k, i_k)$$
$$\text{mit } 0 \leq i \leq |x| + 1 \text{ und } 0 \leq i_j \leq |\alpha_j| \text{ für } j = 1, ..., k$$

eine Konfiguration von M.
Die **Speicherplatzkomplexität von C** ist

$$\mathbf{Space_M(C)} = \max\{|\alpha_i| \mid i = 1, ..., k\}.$$

Space

Sei $C_1, C_2, ..., C_l$ die Berechnung von M auf x. Die **Speicherplatzkomplexität von M auf x** ist

$$\mathbf{Space_M(x)} = \max\left\{\mathrm{Space}_M(C_i) \mid i = 1, ..., l\right\}.$$

Die **Speicherplatzkomplexität von M** ist die Funktion $\mathrm{Space}_M : \mathbb{N} \to \mathbb{N}$, definiert durch

$$\mathbf{Space_M(n)} = \max\left\{\mathrm{Space}_M(x) \mid x \in \Sigma^n\right\}.$$

Space

Bemerkungen

1. Länge des Eingabewortes, hat keinen Einfluss auf die Speicherplatzkomplexität.
2. Mächtigkeit des Alphabets hat keinen Einfluss auf die Speicherplatzkomplexität.

Lemma 6.1

Sei $k \in \mathbb{N} \backslash \{0\}$. Für jede k-Band-TM A, die immer hält, existiert eine äquivalente 1-Band-TM B, so dass

$$\text{Space}_B(n) \leq \text{Space}_A(n)$$

Beweisskizze:

Gleiche Konstruktion wie in Lemma 4.2.

Lemma 4.2 = "Für jede MTM A existiert eine äquivalente TM B".

Wir sehen, dass B genau so viele Felder braucht, wie A.

Lemma 6.2

Zu jeder MTM A existiert eine äquivalente MTM B mit

$$\mathrm{Space}_B(n) \leq \frac{\mathrm{Space}_A(n)}{2} + 2$$

Beweisskizze:

Wir fassen jeweils 2 Felder von A zu einem Feld in B zusammen. $\Gamma_B = \Gamma_A \times \Gamma_A$.
Wir addieren 1 für das ¢ am linken Rand und 1 für das Aufrunden im Fall von ungerader Länge.