Das Postsche Korrespondenzproblem

Eine Instanz des *PKP* ist eine Liste von Paaren aus $\Sigma^* \times \Sigma^*$:

$$(v_1, w_1), \ldots, (v_n, w_n)$$

Eine Lösung ist eine Folge $i_1, \ldots, i_k$ von Indizes $1 \leq i_j \leq n$ mit

$$v_{i_1} \cdot \ldots \cdot v_{i_k} = w_{i_1} \cdot \ldots \cdot w_{i_k}$$

Das Postsche Korrespondenzproblem *PKP*:

Gegeben: Eine Instanz des *PKP*.
Frage: Gibt es eine Lösung?

Dieses Problem *PKP* ist unentscheidbar!

In diesem Kapitel werden wir praxisnähere unentscheidbare Probleme kennenlernen, die kontextfreie Grammatiken betreffen. Als Zwischenschritt betrachten wir das Postsche Korrespondenzproblem, dass von dem deutschen Logiker Emil Post eingeführt wurde.

Erstes Beispiel :

i	v_i	w_i
1	1	111
2	10111	10
3	10	0

Diese Instanz des PKP hat die Lösung $2, 1, 1, 3$, da

$$v_2 v_1 v_1 v_3 = 101111110 = w_2 w_1 w_1 w_3$$

Zweites Beispiel :

i	v_i	w_i
1	10	101
2	011	11
3	101	011

Diese Instanz des PKP hat keine Lösung, wie man leicht einsieht, wenn man eine Lösung zu konstruieren versucht. Jede Lösung müsste mit 1 beginnen, danach kann nur 3 kommen, weil der rechte String um genau eine 1 länger ist als der linke. Danach ist man aber in genau derselben Situation, es kann also keine endliche Lösung geben.

Beweis der Unentscheidbarkeit

Das modifizierte Postsche Korrespondenzproblem *MPKP*:

Gegeben: Eine Instanz des PKP.

Frage: Gibt es eine Lösung $i_1, \ldots, i_k$ mit $i_1 = 1$?

Lemma

$H \leq MPKP$

Lemma

$MPKP \leq PKP$

Das zweite Beispiel oben hat keine Lösung als MPKP, da es schon als PKP unlösbar ist. Das erste Beispiel hat ebenfalls keine Lösung als MPKP, obwohl es eine Lösung hat, da jede Lösung mit 2 beginnen muss.

Aus dem erstem Lemma folgt, dass MPKP unentscheidbar ist, deshalb folgt aus dem zweiten Lemma, dass auch das ursprüngliche PKP unentscheidbar ist.

Reduktion von MPKP auf PKP

Sei $(v_1, w_1), \ldots, (v_n, w_n)$ eine Instanz I des PKP mit Alphabet Σ.

Konstruiere daraus eine neue Instanz I'

$$(v_0', w_0'), (v_1', w_1'), \ldots, (v_n', w_n'), (v_{n+1}', w_{n+1}')$$

mit Alphabet $\Sigma \cup \{*, \$\}$.

Ist $v_i = a_1 a_2 \ldots a_m$ und $w_i = b_1 b_2 \ldots b_\ell$, dann ist

$$v_i' = a_1 * a_2 * \ldots a_m *$$

$$w_i' = * b_1 * b_2 \ldots * b_\ell$$

für $i = 1, \ldots, n$.

Ferner ist $v_0' = * v_1'$ und $w_0' = w_1'$,

sowie $v_{n+1}' = \$$ und $w_{n+1}' = *\$$.

Dann gilt: I' hat eine Lösung gdw. I hat eine Lösung mit $i_1 = 1$.

Auf das erste Beispiel oben angewendet ergibt die Reduktion:

i	v_i'	w_i'
0	$*1*$	$*1*1*1$
1	$1*$	$*1*1*1$
2	$1*0*1*1*1*$	$*1*0$
3	$1*0*$	$*0$
4	$\$$	$*\$$

Jede Lösung dieser Instanz müsste mit 0 beginnen, weil nur die Strings v_0' und w_0' einen gemeinsamen Anfang haben. Dies würde dann aber zu einer Lösung der Ausgangsinstanz führen, die mit 1 beginnt.

Reduktion des Halteproblems auf *MPKP*

Sei $M = (Q, \Sigma, \Gamma, \square, \delta, q_0, F)$ eine DTM und $w \in \Sigma^*$.

Annahmen über M:

- $\delta(q, a)$ ist undefiniert gdw. $q \in F$
- M schreibt kein Leerzeichen $\square$

Wir konstruieren eine Instanz $I(M, w)$ des *MPKP* mit Alphabet $\Gamma \cup Q \cup \{\#\}$, so dass:

$I(M, w)$ hat eine Lösung gdw. M hält bei Eingabe w

Die Annahmen über die Maschine stellen keine Einschränkung dar, wenn man nur daran interessiert ist, ob sie hält oder nicht. Zu jeder DTM M kann man M' konstruieren, die den Einschränkungen genügt, und die bei jeder Eingabe w hält genau dann wenn M bei Eingabe w hält:

- füge einen neuen Haltezustand q_f hinzu und setze $F = \{q_f\}$. Ist $\delta(q, a)$ undefiniert, so ändere dies ab zu $\delta(q, a) = (q_f, a, R)$. Für q_f werden keine Übergänge definiert.
- Füge ein neues Zeichen B zu Γ hinzu. In jedem Übergang, wo M ein $\square$ schreibt, ersetze dies durch B. Schließlich setze für jeden Zustand $\delta(q, B) = \delta(q, \square)$, so dass sich M' beim Lesen von B genauso verhält wie M beim Lesen von $\square$.

Die Idee der Reduktion ist, dass jede Lösung von $I(M, w)$ aus der Folge der Konfigurationen von M bei Eingabe w besteht, jeweils durch $\#$ getrennt. Ist also die Berechnung von M bei Eingabe w

$$q_0 w = \alpha_1 \vdash_M \alpha_2 \vdash_M \ldots \vdash_M \alpha_t$$

dann erzeugt eine Lösung von $I(M, w)$ den String

$$\#\alpha_1\#\alpha_2\#\ldots\#\alpha_t\#$$

Reduktion des Halteproblems auf *MPKP*

Die Instanz $I(M, w)$ enthält die folgenden Paare:

v_i	w_i	Erklärung
$\#$	$\#q_0w\#$	das erste Paar
a	a	für $a \in \Gamma$
$\#$	$\#$	
qa	bp	für $\delta(q, a) = (p, b, R)$
$qa\#$	$bp\square\#$	"
cqa	pcb	für $\delta(q, a) = (p, b, L)$ und $c \in \Gamma$
$\#qa$	$\#p\square b$	für $\delta(q, a) = (p, b, L)$
qa	q	für $q \in F$ und $a \in \Gamma$
aq	q	"
$q\#\#$	$\#$	für $q \in F$

Das erste Paar, mit der eine Lösung beginnen muss, schreibt zuerst die Startkonfiguration von M in die rechte Seite. Die weiteren Paare sind so gestaltet, dass beim kopieren dieser Konfiguration auf die linke Seite in der rechten Seite die nächste Folgekonfiguration erzeugt wird.

Mit den Paaren in der 2.-3. Zeile werden die Bandsymbole und Trennzeichen kopiert, die sich zur nächsten Konfiguration hin nicht ändern.

Die Paare in der 4.-7. Zeile sorgen dafür, dass beim Kopieren des Bandstücks um den Zustand in der rechten Seite der Übergang von M nachgebildet wird. Dabei ist die 4.-5. Zeile für Übergänge mit Kopfbewegung nach rechts, davon wiederum Zeile 5 für den Spezialfall, wo der Kopf ganz rechts steht. Analog sind Zeilen 6-7 für Übergänge, wo sich der Kopf nach links bewegt.

Die Paare in Zeilen 8-9 bewirken, dass im Haltezustand die Bandsymbole links und rechts vom Kopf verschwinden, so dass ganz am Schluss der Lösung nur noch der Endzustand dasteht. Dann kann das letzte Paar in Zeile 10 die Lösung abschließen.

Es ist nun klar, dass jede haltende Berechnung von M so zu einer Lösung führt, und umgekehrt jede Lösung einer haltenden Berechnung von M entsprechen muss. Daher ist die Konstruktion eine Reduktion von H auf *MPKP*.

Disjunktheit kontextfreier Sprachen

Das Disjunktheitsproblem *DISJ*:

Gegeben: Zwei kfG G_1 und G_2
Frage: Ist $L(G_1) \cap L(G_2) = \emptyset$?

Satz

Das Disjunktheitsproblem für kfG ist unentscheidbar.

Beweis: Durch Reduktion $PKP \leq DISJ$

Dies ist nur ein erstes Beispiel eines Problems, dass in der Informatik vorkommende Konzepte betrifft und unentscheidbar ist. Im weiteren werden wir noch mehr solche unentscheidbaren Probleme über kontextfreie Grammatiken sehen. Unentscheidbare Probleme treten aber überall in der Informatik auf, wo man mit ausdrucksstarken Konzepten umgehen muss.

Reduktion von *PKP* auf *DISJ*

Sei $(v_1, w_1), \ldots, (v_n, w_n)$ eine Instanz I des PKP mit Alphabet Σ.

Definiere zwei kfG mit Alphabet $\Sigma' = \Sigma \cup \{1, \ldots, n\}$,

$G_1(I) = (\{A\}, \Sigma', P_1, A)$ und $L_1(I) = L(G_1(I))$
$G_2(I) = (\{B\}, \Sigma', P_2, B)$ und $L_2(I) = L(G_2(I))$

P_1 enthält Produktionen:

$A \to v_i A\, i \quad$ und $\quad A \to v_i\, i \quad$ für $1 \leq i \leq n$

P_2 enthält Produktionen:

$B \to w_i B\, i \quad$ und $\quad B \to w_i\, i \quad$ für $1 \leq i \leq n$

Dann gilt: $L_1(I) \cap L_2(I) \neq \emptyset$ genau dann, wenn I eine Lösung hat.

Die Idee der Reduktion ist: die Grammatik $G_1(I)$ erzeugt die Wörter, die aus der linken Seite der PKP-Instanz erzeugt werden, zusammen mit der entsprechenden Indexfolge, also die Wörter der Form $v_{i_1} v_{i_2} \ldots v_{i_k} i_k \ldots i_2 i_1$, und $G_2(I)$ entsprechend für die rechte Seite.

Ein Wort wird also dann und nur dann von beiden Grammatiken erzeugt, wenn für die darin stehende Indexfolge die linken und rechten Seiten den gleichen String erzeugen, wenn also I eine Lösung hat.

Daher reduziert die Konstruktion das PKP auf das Disjunktheitsproblem.

Mehrdeutigkeit

Definition: Eine kfG G ist eindeutig, wenn es für jedes Wort $w \in L(G)$ genau einen Ableitungsbaum gibt.

Sonst heißt G mehrdeutig.

Satz

Das folgende Probem ist unentscheidbar:

Gegeben: *Eine kfG G*

Frage: *Ist G mehrdeutig?*

In den Grammatiken $G_1(I)$ und $G_2(I)$ gibt es für jedes Wort höchstens eine Ableitung, sie sind also eindeutig. Nun betrachtet man die Grammatik, die entsteht, wenn $G_1(I)$ und $G_2(I)$ vereinigt werden, und ein neues Startsymbol S hinzugefügt wird, mit den Produktionen $S \to A$ und $S \to B$.

Dann gilt: I hat eine Lösung gdw. es gibt ein Wort w, dass aus $G_1(I)$ und aus $G_2(I)$ hergeleitet werden kann. Dieses Wort w hat dann zwei Ableitungen in der vereinigten Grammatik. Andernfalls hat jedes Wort höchstens eine Ableitung.

Die Konstruktion reduziert also das PKP auf das Mehrdeutigkeitsproblem.

Ein bekannter Fall, bei dem die Mehrdeutigkeit von Grammatiken problematisch ist, tritt beim `if-then-else` Konstrukt auf. Angenommen, in einer Sprache soll diese Konstrukt mit optionalem `else` verwendet werden. Dann wäre es natürlich, dafür 2 Produktionen einzuführen:

$A \to$ `if` B `then` A `else` A

$A \to$ `if` B `then` A

Dann hätte aber der folgende String zwei wesentlich verschieden Ableitungen, die auch zu unterschiedlicher Semantik führen, weil nicht klar ist, zu welchem der `if` das `else` gehört:

`if` B `then if` B `then` A `else` A

Weitere Probleme

Lemma

Auch die Sprachen $\overline{L_1(I)}$ und $\overline{L_2(I)}$ sind kontextfrei.

Seien G_1 und G_2 kfG, und R ein regulärer Ausdruck.

Folgende Probleme sind unentscheidbar:

- Ist $L(G_1) = L(G_2)$?
- Ist $L(G_1) = L(R)$?
- Ist $L(G_1) \subseteq L(G_2)$?
- Ist $L(R) \subseteq L(G_1)$?

Wir beweisen hier nicht, dass die Sprachen $\overline{L_1(I)}$ und $\overline{L_2(I)}$ kontextfrei sind,, am einfachsten sieht man dies durch Konstruktion von PDA für diese Sprachen.

Sei $G_1'(I)$ eine Grammatik für die Sprache $\overline{L_1(I)} \cup \overline{L_2(I)} = \overline{L_1(I) \cap L_2(I)}$, und G_2' eine Grammatik mit $L(G_2') = (\Sigma')^*$. Da der Durchschnitt $L_1(I) \cap L_2(I)$ nicht leer ist genau dann, wenn I eine Lösung hat, ist $L(G_1'(I)) \neq L(G_2')$ genau dann, wenn I eine Lösung hat. Die Konstruktion reduziert also PKP auf das Gleichheitsproblem.

Da $(\Sigma')^*$ eine reguläre Sprache ist, kann G_2' auch durch einen regulären Ausdruck dafür ersetzt werden, was die zweite Aussage zeigt.

Da ferner für jede Sprache L über Σ' gilt $(\Sigma')^* = L$ genau dann wenn $(\Sigma')^* \subseteq L$, gelten sogar die dritte und vierte Aussage.